



National Library
of Canada

Acquisitions and
Bibliographic Services Branch

395 Wellington Street
Ottawa, Ontario
K1A 0N4

Bibliothèque nationale
du Canada

Direction des acquisitions et
des services bibliographiques

395, rue Wellington
Ottawa (Ontario)
K1A 0N4

Your file Votre référence

Our file Notre référence

NOTICE

The quality of this microform is heavily dependent upon the quality of the original thesis submitted for microfilming. Every effort has been made to ensure the highest quality of reproduction possible.

If pages are missing, contact the university which granted the degree.

Some pages may have indistinct print especially if the original pages were typed with a poor typewriter ribbon or if the university sent us an inferior photocopy.

Reproduction in full or in part of this microform is governed by the Canadian Copyright Act, R.S.C. 1970, c. C-30, and subsequent amendments.

AVIS

La qualité de cette microforme dépend grandement de la qualité de la thèse soumise au microfilmage. Nous avons tout fait pour assurer une qualité supérieure de reproduction.

S'il manque des pages, veuillez communiquer avec l'université qui a conféré le grade.

La qualité d'impression de certaines pages peut laisser à désirer, surtout si les pages originales ont été dactylographiées à l'aide d'un ruban usé ou si l'université nous a fait parvenir une photocopie de qualité inférieure.

La reproduction, même partielle, de cette microforme est soumise à la Loi canadienne sur le droit d'auteur, SRC 1970, c. C-30, et ses amendements subséquents.

UNIVERSITY OF ALBERTA

Interaction Techniques for Solid Modeling with a 3D Input Device

BY

Jiandong Liang



A thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

DEPARTMENT OF COMPUTING SCIENCE

Edmonton, Alberta
Fall 1995



National Library
of Canada

Acquisitions and
Bibliographic Services Branch

395 Wellington Street
Ottawa, Ontario
K1A 0N4

Bibliothèque nationale
du Canada

Direction des acquisitions et
des services bibliographiques

395, rue Wellington
Ottawa (Ontario)
K1A 0N4

Your file Votre référence

Our file Notre référence

THE AUTHOR HAS GRANTED AN
IRREVOCABLE NON-EXCLUSIVE
LICENCE ALLOWING THE NATIONAL
LIBRARY OF CANADA TO
REPRODUCE, LOAN, DISTRIBUTE OR
SELL COPIES OF HIS/HER THESIS BY
ANY MEANS AND IN ANY FORM OR
FORMAT, MAKING THIS THESIS
AVAILABLE TO INTERESTED
PERSONS.

L'AUTEUR A ACCORDE UNE LICENCE
IRREVOCABLE ET NON EXCLUSIVE
PERMETTANT A LA BIBLIOTHEQUE
NATIONALE DU CANADA DE
REPRODUIRE, PRETER, DISTRIBUER
OU VENDRE DES COPIES DE SA
THESE DE QUELQUE MANIERE ET
SOUS QUELQUE FORME QUE CE SOIT
POUR METTRE DES EXEMPLAIRES DE
CETTE THESE A LA DISPOSITION DES
PERSONNE INTERESSEES.

THE AUTHOR RETAINS OWNERSHIP
OF THE COPYRIGHT IN HIS/HER
THESIS. NEITHER THE THESIS NOR
SUBSTANTIAL EXTRACTS FROM IT
MAY BE PRINTED OR OTHERWISE
REPRODUCED WITHOUT HIS/HER
PERMISSION.

L'AUTEUR CONSERVE LA PROPRIETE
DU DROIT D'AUTEUR QUI PROTEGE
SA THESE. NI LA THESE NI DES
EXTRAITS SUBSTANTIELS DE CELLE-
CI NE DOIVENT ETRE IMPRIMES OU
AUTREMENT REPRODUITS SANS SON
AUTORISATION.

ISBN 0-612-06248-1

Canada

UNIVERSITY OF ALBERTA

RELEASE FORM

NAME OF AUTHOR: Jiandong Liang

TITLE OF THESIS: **Interaction Techniques for Solid Modeling with a 3D Input Device**

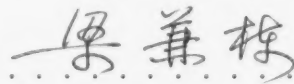
DEGREE: Doctor of Philosophy

YEAR THIS DEGREE GRANTED: 1995

Permission is hereby granted to the University of Alberta Library to reproduce single copies of this thesis and to lend or sell such copies for private, scholarly or scientific research purposes only.

The author reserves all other publication and other rights in association with the copyright in the thesis, and except as hereinbefore provided neither the thesis nor any substantial portion thereof may be printed or otherwise reproduced in any material form whatever without the author's prior written permission.

(Signed)



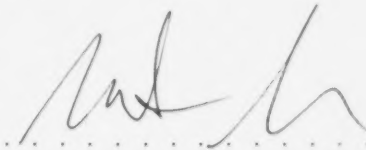
Jiandong Liang
32-1055 72 Ave. N.W., Calgary,
Alberta, Canada T2K 5S4

Date: *May 27, 1995*

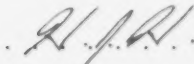
UNIVERSITY OF ALBERTA

FACULTY OF GRADUATE STUDIES AND RESEARCH

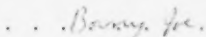
The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research for acceptance, a thesis entitled **Interaction Techniques for Solid Modeling with a 3D Input Device** submitted by Jiandong Liang in partial fulfillment of the requirements for the degree of Doctor of Philosophy.



Dr. M. W. Green (Supervisor)



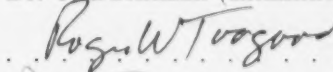
Dr. R. Pausch (External)



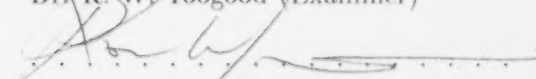
Dr. B. Joe (Examiner)



Dr. J. Buchanan (Examiner)



Dr. R. W. Toogood (Examiner)



Dr. D. Rochfort (Examiner)

Date: May 5, 1995

To my parents

Abstract

Modeling in three dimensional (3D) space is an important field in computer-aided design (CAD) and computer graphics. The use of computers in the design of 3D models has been fruitful, yet largely confined to the late stages of the design process, namely, analysis, simulation, drafting, and documentation. The early stage — that of conceptual design, or sketching — has so far hardly been touched. The user interfaces of most CAD systems impose a limit on the speed that a designer can translate his or her ideas into a computer model. As a result, designers usually prefer a more facile medium when sketching. In most cases, this means sketching with pencil on paper.

This thesis originated from the vision that CAD technologies can be extended into the sketching stage, to become a tool covering the entire design process. We focused on the ergonomics of interface design, since several problems in this area were identified as major limiting factors. New interaction techniques were developed to provide the designer with easy and flexible ways of creating and manipulating 3D shapes. With a 3D input device and a fast graphics workstation, an environment was developed in which a designer can directly interact with 3D shapes using an array of new interaction techniques. In particular, a non-immersive 3D interface style for the modeling system was developed allowing a smooth transition between sketching and later design stages, and the integration of existing 2D interaction techniques. New interaction techniques at various levels were developed. First, we designed basic interface elements such as 3D menus, selectors, and dials. We then developed techniques for the creation, placement and reshaping of 3D shapes. We also developed inspection tools for 3D shapes. These new interaction techniques are integrated into

a prototype solid modeling system (JDCAD) for study and evaluation.

We informally evaluated the overall effectiveness of the new interface style and interaction techniques by first comparing our prototype system JDCAD with a commercial CAD system and then with hand sketching. It was shown that it took less time for a designer to create test objects in JDCAD than in the commercial CAD system. We also showed that for an expert user, the sketching speed with JDCAD is comparable with hand sketching using pencil and paper. These encouraging results demonstrate the potential of the new interaction techniques, and indicate a significant progress in extending the use of CAD systems into the sketching stage of 3D modeling.

Acknowledgements

I would like to express my sincere thanks to Dr. Mark Green, my supervisor, for his encouragement and support throughout the course of my research. I would also like to thank the members of my committee Drs. Barry Joe, John Buchanan, Roger Toogood, Desmond Rochfort, and Randy Pausch, for their valuable and insightful comments. I am very grateful to Dr. Walter Bischof for his comments on early versions of the thesis, and also to Ken Hinckley at University of Virginia who helped reviewing the thesis.

Thanks go to the graphics research group at the University of Alberta for many interesting discussions. I am especially grateful to Chris Shaw for many helpful discussions.

To University of Alberta, Faculty of Graduate Studies and Research, and Department of Computing Science, for technical and financial support.

To my friends Haiying Wang, Chenghui Wang, Guanghong Zhang, and many others, for their support and friendship.

To my dear wife Hui, for her patience, encouragement, support, and love that gave me the strength to pursue my dreams.

To my daughter Jenny, who fills me with love and wonder.

Finally, to my parents Guanghua Liang and Rongxiu Zhang, for their love and support throughout my life. This thesis is for them.

Contents

1	Introduction	1
1.1	The challenge of 3D modeling	1
1.2	Terminology	3
1.3	Research plan	3
1.4	Contributions	4
1.5	Non-contributions	5
1.6	Outline of the dissertation	5
2	Motivations	7
2.1	Initial drive	7
2.2	A broader view	8
2.3	Observations	11
2.4	New interaction style	12
2.5	Chapter summary	13
3	Related Work	14
3.1	3D interaction techniques based on 2D pointing devices	14
3.1.1	Conventional approaches	15
3.1.2	Recent advances	17
3.1.3	Section summary	25
3.2	High-DOF interaction techniques	26
3.2.1	Immersive style	27
3.2.2	Non-immersive style	29
3.2.3	Section summary	32
3.3	Chapter summary	32
4	Interaction Style — The Design Philosophy	34
4.1	Overall set up	36
4.2	3D interaction techniques	39
4.3	Other issues	39
4.4	Chapter summary	40

5	JDCAD — A Prototype Implementation	41
5.1	Brief history	41
5.2	3D input device	42
5.3	3D display	45
5.4	3D input processing	46
5.4.1	Filtering	47
5.4.2	Clutching	47
5.5	Chapter summary	48
6	Menus, Selectors, and Dials	49
6.1	Menu selection	49
6.1.1	Daisy menu	50
6.1.2	Ring menu	51
6.1.3	Implementation notes	53
6.2	Object selection	54
6.2.1	From laser gun to spotlight	54
6.2.2	Unification with 2D metaphor	56
6.2.3	Implementation notes	57
6.2.4	Other feedback options	57
6.3	Graphic dials	58
6.4	Chapter summary	60
7	Modeling Operations	61
7.1	Creation	62
7.2	Placement	63
7.2.1	Position control	63
7.2.2	Orientation control	68
7.2.3	Section summary	70
7.3	Reshaping	70
7.3.1	Region-based reshaping	71
7.3.2	Support of precision	73
7.4	Bat-based clipping	74
7.5	Chapter summary	75
8	Evaluating JDCAD	76
8.1	Comparing with a commercial CAD system	76
8.1.1	The challenges	76
8.1.2	The first test object	77
8.1.3	Using a 2D-based System	78
8.1.4	Using JDCAD	81
8.1.5	The second test object	87
8.1.6	Discussion of the timing results	88
8.1.7	Section summary	90
8.2	Comparing with hand sketching	91

8.2.1	Designing the experiment	91
8.2.2	The test objects	91
8.2.3	Discussion of timing results	93
8.2.4	Section summary	97
8.3	Chapter summary	98
9	Applications of JDCAD	99
9.1	VR demos	99
9.2	Other applications	102
9.3	Chapter summary	104
10	Conclusions and Future Work	105
10.1	Summary	105
10.2	Limitations	107
10.3	Future topics	108
10.4	Other related topics	109
10.5	Concluding remarks	111
A	JDCAD	112
A.1	Introduction to JDCAD	112
A.1.1	Object type and hierachy	112
A.1.2	States and roles	112
A.1.3	Active axis	113
A.2	List of Commands	113
A.2.1	Creation of primitive shapes	114
A.2.2	Selecting objects	114
A.2.3	Selecting axis	115
A.2.4	Placement of objects	115
A.2.5	Transformation of model	116
A.2.6	Reshaping of objects	116
A.2.7	Alignment and conformation	116
A.2.8	Set and group operations	118
A.2.9	Cursor control	119
A.2.10	Viewing and rendering	119
A.2.11	General editing	120
A.2.12	File operations	120
A.2.13	Buttons	121
A.2.14	Pull-down menus	121
A.2.15	Miscellaneous	121
A.3	Summary	121
B	Hand Sketches	122
B.1	Subject No. 1.	122
B.2	Subject No. 2.	123

B.3 Subject No. 3.	124
B.4 Subject No. 4.	125
B.5 Subject No. 5.	126
B.6 Subject No. 6.	127
B.7 Subject No. 7.	128
B.8 Subject No. 8.	129
B.9 Subject No. 9.	130
B.10 Subject No. 10.	131

Bibliography	133
---------------------	------------

List of Figures

1	A 3D model of a mechanical bracket.	9
2	A Skitter moves on the surface of an extrusion. The two dimensional cursor is not shown. (a) on front surface. (b) on back surface.	18
3	Aligning two objects with jacks. (a) two jacks are placed on the body and handle of a mug. (b) after aligning the two handles.	18
4	Triad mouse technique.	21
5	Grips on a 3D cursor.	22
6	Multiple hit-testing through nonparallel views.	23
7	Arc interpretation.	25
8	A cutaway view of Schmandt's workstation system.	30
9	Using 3D input device to position an articulated figure.	31
10	Overall setting of the style.	37
11	Top: The button-enhanced bat. Middle and bottom: Four postures for holding the bat.	43
12	A snapshot of a pop-up daisy.	51
13	Directions associated with the Isotrak sensor.	52
14	A snapshot of a ring menu.	53
15	A snapshot of spotlight selection.	56
16	A cross section of the selection cone showing isodistance surfaces of the anisotropic metric S . Point P is considered closer to the origin than point Q	57
17	A bat-operated dial is being used to adjust the spread angle of a cone.	58
18	A secondary dial is used for higher precision.	59
19	Active snap grid.	66
20	Region-based reshaping.	72
21	A dialogue box for shape editing.	73
22	A clipping tool is being used to explore the internal structure of a casing.	74
23	The first test object.	78
24	Choosing the cylinder primitive with the ring menu.	82
25	First Cylinder created.	82

26	Second cylinder created and aligned to the first one.	83
27	A pipe formed by "subtract" operation.	84
28	A box formed and centered to the pipe with respect to the selected axis.	85
29	A plate created and aligned to the object.	86
30	A third cylinder is being positioned with active snap grid.	86
31	The second test object in AutoCAD wireframe mode.	87
32	Comparison of drawing times.	89
33	Test object A.	92
34	Test object B.	92
35	Test object C from two different views.	93
36	A sample sketch of the test object A.	93
37	A sample sketch of the test object B.	94
38	A sample sketch of the test object C.	94
39	Comparison between hand sketching and using JDCAD.	95
40	A snapshot from a dinosaur scene.	100
41	A biplane built in JDCAD+.	100
42	A knight riding a bug.	101
43	A helicopter designed in JDCAD+.	101
44	Illustrating a culling problem.	102

List of Tables

8.1	Timing for the first test object.	88
8.2	Timing for the second test object.	88
8.3	Hand sketching and JDCAD time (in seconds).	94

Chapter 1

Introduction

1.1 The challenge of 3D modeling

Throughout the history of computer graphics and computer-aided design (CAD), the modeling of three dimensional (3D) forms has always been a very important topic of research and development. The fact that people live in a 3D world, and perceive it as such, propels the unending quest for the ability to create, modify, visualize and manipulate 3D objects by computers.

Yet, it remains a very challenging task. As Robert Sproull put it in [40]: “What remains hard is modeling. ... The grand challenges in three dimensional graphics are to make simple modeling easy and to make complex modeling accessible to far more people.” The same proposition has also been made again and again in the CAD community [1, 7, 14, 18, 28, 29].

In order to understand the problem, we need to look at a broader picture.

In general, the process of geometric design is a transformation of intangible concepts (mental images) into tangible forms (graphical drawings or physical models). As such, the properties of the underlying medium greatly influence the design process. Important properties include dimensionality, method of initial recording and modification, accommodation of accuracy, support of visualization and manipulation.

Before computer technology was available, people mainly relied on paper medium

for geometric design. It has the advantage of easy recording (with a pencil or pen), and it supports limited modification. Medium levels of accuracy are supported by scaled drawing, and high levels of accuracy are supported by text-based dimensioning. However, paper medium has several disadvantages. First, modifications are difficult to perform, as it takes a much longer time to erase markings than to draw them. Second, the paper is a 2D medium, thus it can only record static projections of a 3D shape.

CAD technology supplies a new medium. This electronic medium offers many advantages over paper. For example, erasing a mark can be done very easily. A true 3D model can be recorded and dynamically visualized. However, it also has its disadvantages. Interaction with a CAD system is less flexible than drawing with pencil and paper, and often at a slower speed. The process of converting paper drawings to a computer model is also time consuming.

By moving from paper to CAD systems, one gains processing power but sacrifices flexibility and agility. This is a major reason behind many difficulties that people encounter when using CAD systems.

When it comes to 3D modeling, the problem is made more acute for both media, by the additional dimension. Paper can only record 2D projections of 3D shapes, due to its 2D nature. In order to work on different parts of an object, one has to draw many different projections. The task of assimilating them into a single 3D mental model is non-trivial. For CAD systems, their internal processing power becomes more useful, as they can represent true 3D models. However, the communications between the designer and CAD systems become the bottleneck, as most CAD systems uses 2D CRT screens for display (output) and 2D pointing devices for input. Because of this bottleneck, the use of CAD systems for 3D modeling is mainly limited to later stages of the design process, where interaction speed is not a critical factor. This leads to the current practice of sketching on paper then transferring to CAD systems for processing. In other words, the disadvantages of paper have to be endured in the early stage, the high cost of transferring them to CAD systems will remain, and the

same for the lack of flexibility in interacting with CAD models, all because there is no better alternative.

The goal of this dissertation is to extend CAD systems, as a medium for 3D modeling, so they can cover more of the early stage of the design process. In this way, the disadvantage of paper medium can be avoided, the costly process of transferring paper drawings to CAD models can be eliminated, and the intrinsic processing power of computer systems can be made available to the designer throughout the entire design process.

The main task is to design a new interface style for 3D solid modeling such that it satisfies the following two criteria: (a) intuitive, flexible, and agile enough to merit being used for sketching, and (b) capable of supporting adequate accuracy to allow a seamless transition into later stages of the design. These criteria are met with the innovative use of a 3D tracking device as a new input device, a careful set up of system configuration, and proper integration with traditional interaction techniques.

1.2 Terminology

The term *3D modeling* in this dissertation denotes the creation and modification of 3D objects from simple regular shapes. While 3D free form surface design is a very important topic in computer-aided geometric design, it is outside the scope of this dissertation. The *manipulation* of an object refers to changing its attributes such as size, position, and orientation.

1.3 Research plan

We first observed that the traditional 2D interaction techniques are inadequate for the task of 3D modeling, and new techniques must be developed to meet the demands. We also noted that many early attempts at developing 3D interaction techniques failed to address various important issues, and thus did not gain general acceptance

into the CAD community.

The research was carried out in the following steps:

1. pitfalls, as well as merits, of previous attempts at facilitating 3D solid modeling were identified,
2. a new type of 3D direct manipulation user interface was proposed,
3. various design issues in implementing the proposed interface were addressed,
4. a prototype system was implemented to test the effectiveness of the proposed interface, and favorable results were obtained,
5. experience with the prototype system was summarized into a general guideline for designing 3D direct interaction,
6. intrinsic limitations of the new interface style were also discovered, and several topics for future research were clearly identified.

1.4 Contributions

As we stated above, this research concerns itself with the issue of user interfaces for CAD systems pertaining to three dimensional solid modeling in the context of industrial design process.

We first discuss user interface design issues that have major impact on the usability of a 3D modeler in early stage of the design process. We lay down the path leading from our design philosophy to a specific interface style- one that couples the use of a single hand-held light-weight six degrees of freedom input device, a sitting posture with elbow resting on the desktop, and a non-immersive kinetic 3D display mechanism. This style retains traditional 2D interaction techniques while accommodating 3D interaction techniques, thus combining their strength in dealing with various challenges in 3D solid modeling.

Second, a collection of interaction techniques are developed in a prototype system, to explore the potential of the new interface style. These interaction techniques range from low level ones such as clutch and filtering methods, to medium level ones such as daisy and ring menus, spotlight selector, and dials, on to high level ones such as multiple snap dragging, animated alignment, orientation snapping, region-based reshaping, clipping tools, etc, in assisting 3D solid modeling. Though they are by no means optimal or complete, they show the advantages and potential of this interface style in 3D solid modeling.

Third, we demonstrate that 3D objects can be sketched with a computer system in comparable ease with that of the pencil-on-paper practice, and the objects so created can be progressively refined and modified in the same style with the same ease. This shows a promising way of extending the coverage of CAD systems into the very early stage of the design process.

From a philosophical point of view, the research is conducted in search of a balanced and harmonic congruence between existing and new technology, and criticizes both radicalism and conservatism in user interface design.

1.5 Non-contributions

This research is not intended to build a full-blown, industrial-strength 3D modeler, as it requires much more resources than those at our disposal.

1.6 Outline of the dissertation

In the following chapter, we will discuss the motivations behind this research. In chapter 3, we will analyze the problems with 3D modeling using traditional interaction techniques, and also the pitfalls in several attempts to employ 3D direct manipulations. Based on these observations, we propose a new interaction style in chapter 4, and discuss the design philosophy. We start to discuss a prototype implementation

of the new interface style in chapter 5. We first give a brief history overview of the system, then explore issues at low level, such as 3D input processing. Chapter 6 is given to the basic interface components: menu selection, object selection, and graphical dials. Chapter 7 discusses the creation, placement, reshaping, and inspection of 3D objects. Chapter 8 presents some preliminary evaluation of the prototype system, as well as attempts at comparing it with existing CAD tools and hand sketching. Chapter 9 discusses some applications of the prototype system. The last chapter summarizes the experiences into guidelines for designing 3D interaction techniques and presents topics for future research. It concludes the dissertation. We include a simple listing of commands in the prototype system in appendix A, and the collection of hand sketches produced during the evaluation test in appendix B.

Chapter 2

Motivations

2.1 Initial drive

The initial drive for building an easy-to-use 3D modeler came from our virtual reality research [27]. In order to study the effect of sensor noise and lag in a virtual environment, one needs to quickly build and modify 3D virtual environments, hopefully in a matter of minutes or hours, but not days. In order to obtain a meaningful measurement for head motion in a virtual environment, the environment itself should bear a reasonable level of realism. For example, a simple grid of 3D lines is not very useful for this purpose. One experimental environment was built as a simplified scene of the Graphics Lab where this research was conducted. The environment consisted of the graphical representation of the following objects:

- interior walls of the lab,
- two door frames,
- five tables,
- six graphics monitors,
- two file cabinets,
- two bookshelves, and

- several equipment boxes.

The ceiling and floor were shown as tiled according to their real appearance. This environment was just minimally acceptable for the study of head movement during target search and focus shift. However, it took the author about a week of time to construct and test. Moving virtual furniture around or adding virtual objects was also time consuming. It occurred to the author that something would have to be done to alleviate such drudgery.

It soon became evident that no commercial products were available for this purpose. Experience of the virtual reality research group at University of North Carolina at Chapel Hill confirmed this [9]. In their experiment with architectural walkthrough, one graduate student spent about six months of time building a 3D model of an existing house, using a commercial CAD program, AutoCAD. While a professional AutoCAD operator could probably do it in a much shorter time, say one tenth of it (which is about three weeks), it is still far slower than desirable. Besides, it takes a long time, on the order of several months, to learn a sophisticated CAD program like AutoCAD. This set off the initial search for a better solution.

2.2 A broader view

When we looked more closely into the field of computer aided design (CAD), the following facts were revealed.

First, CAD systems are being widely used in engineering designs, especially mechanical and civil engineering. Second, if we use any of the usual categorization schemes for the phases of the design activity, we find that the proliferation of such CAD systems applies only to the later, and hardly ever to the earliest phases [25].

The simplest taxonomy has four phases:

- conceptual, creative,
- analysis, simulation,

- detailing, drafting,
- documentation.

It is a familiar fact that the area of fastest penetration was in the third phase (computer-aided drafting) to be followed by progress both “up” and “down” the list, towards such techniques as finite-element methods on the one hand, and the computer-aided generation of numerical control programs on the other. The use of a more detailed categorization does not alter the message [14, 18, 25].

The first phase — that of creative, innovative, conceptual design — has so far hardly been touched.

If we focus ourselves to the realm of 3D design, the lack of quick prototyping tools is even more acute.

With a commercial CAD system, it usually takes a few hours to several days to create a 3D model of moderate complexity. For example, Hodkinson and Patel showed that the 3D model of a mechanical bracket in figure 1 would take several hours to create in a CAD system [20]. However, a designer often wants to sketch a prototype in a few minutes [25]. Such a significant discrepancy makes it anything but possible to sketch with such a system.

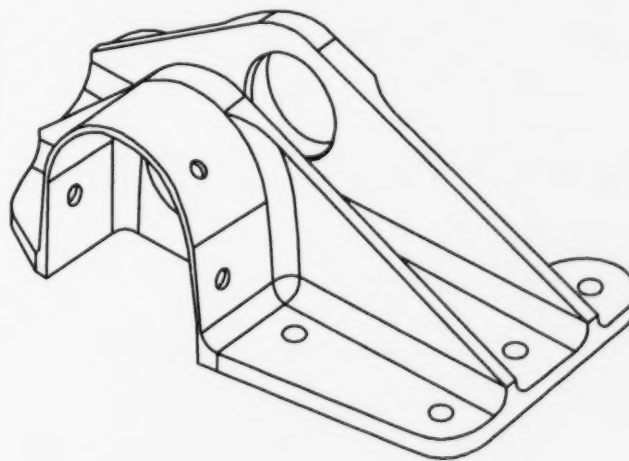


Figure 1: A 3D model of a mechanical bracket.

Is there a need for quick prototyping? The answer is positive. It is common sense

that the earlier stages play more important roles in determining the final quality of the design. The earlier a design mistake can be identified, the less effort it will take to correct it. By increasing the efficiency of the initial stage, one can allow for more evaluation-revision cycles to be made before committing a design for later stages, thus eliminating more deficiencies. Alternatively, one can shorten the design period while maintaining the same quality.

The discrepancy between the need for quick prototyping and the lack of such support from CAD systems resulted in the current practice of a hybrid approach. The sketching phase is usually done with pencil and paper [21, 24, 25]. When the design is reasonably stable and detailed, it is transferred to a CAD system for later phases such as analysis and simulation.

While paper is preferred in the sketching phase, it is by no means an ideal medium. First of all, it is 2D by nature, and can only record projections of 3D objects. Even simple visualization processes such as projection (whether parallel or perspective) and hidden line removal have to be performed consciously. When a different projection is needed, another sketch has to be made. Modification is also time consuming, because the paper medium is asymmetric with respect to marking and erasing — it is much easier and faster to draw a line than to erase it. As the paper medium has limited resolution, the designer often has to record details in separate drawings. This increases the cognitive load to visualize a design in its entirety. These disadvantages do not exist for CAD systems. With a CAD system, the designer can erase a line easier than drawing it. Projection and hidden line removal can be performed automatically. Zooming and panning are also easily supported. Nevertheless, paper is still the medium of choice for sketching.

This prompted the author to seek the reason behind it.

2.3 Observations

The long turnaround time required for creating a 3D model in a conventional CAD system indicates a lack of ease in using such a modeling system. While there are several reasons behind this, such as the lack of support for constraints [33], the emphasis of processing power over expressing power [25], etc., one prominent reason is with the user interface.

Contemporary CAD systems are almost invariably based on two dimensional interaction techniques, as characterized by their use of 2D digitizers (such as mice or tablets) as the main input devices. While such settings are suitable for certain tasks such as detailed drafting or capturing of paper drawings, they are not suitable for quick sketching. Although the CRT screen looks not too different from a piece of paper, the fundamental difference between a computer and the human perception system makes it impossible for the machine to interpret a 2D sketch and to build a 3D model from it. In other words, complete information about the 3D model has to be entered into the computer. This includes the 3D position and orientation of various features of the model.

In general, a 3D solid has six degrees of freedom, and needs six parameters to completely define its position and orientation in space. However, a 2D digitizer can only supply two parameters at a time. This discrepancy forces the designer to manipulate a 3D object in one subspace at a time. This created three major problems.

First, the user has to define the mapping from the 2D subspace into the 3D space, before any data input can take place. However, this is not an easy task given the 2D nature of the interaction. Usually, the user has to use numeric input from the keyboard to specify a point not lying in the current 2D subspace. This only provides a point to establish some construction lines from where a reference coordinate system (i.e. the mapping from 2D to 3D) can then be specified. As we will note later in the discussion of experimental results, the specification of reference coordinate systems takes a large portion of the total 3D modeling time.

Second, since the interaction is restricted to a 2D subspace, the user is forced to

decompose a 3D task into 2D sub-tasks. For example, a box has to be first created as a 2D rectangle and then grown in the third dimension. Such decomposition of tasks lowers the level of abstraction, and slows down the modeling process.

Third, 3D rotation poses a difficult challenge. The fact that rotations are not commutative is the fundamental obstacle to decomposition-style interaction. When using 2D interactions, a 3D rotation has to be specified by at least two steps. Different orders in executing the steps will result in different rotations. This is often quite counter-intuitive. Furthermore, in order to cancel a rotation specified in steps, these steps have to be undone in the reverse order. This also adds to the difficulty in manipulating 3D orientations.

In a word, the limitation of 2D interaction makes 3D modeling a difficult task, and slows it down.

2.4 New interaction style

Recent years have seen the emergence of a new interaction style often termed *virtual reality*. The main characteristics of a virtual reality interface is that the user is placed in a computer generated 3D visual environment, where he or she can interact with the environment in a natural manner (such as by hand or body motion). The advantage of such a interface derives from the fact that people are familiar with 3D environments — the real world which they live in is one. By placing the user into a 3D environment, one taps a human being's inherent ability of 3D perception and motor control, which have been perfected through a long period of evolution. For example, to move a virtual object within hand's reach, one can simply reach out to grab the object, carry it through space and set it down. Such an operation requires little conscious planning even with the presence of a modest amount of obstacles.

These novel interaction techniques seem to provide a promising alternative to the traditional 2D interaction style, as they broaden the communication channel between the user and the system, and have the potential of eliminating the various constraints

introduced by 2D input devices. From a different point of view, 3D modeling appears to be a suitable application for virtual reality technology.

This duality is one of the motivations behind this research.

2.5 Chapter summary

The motivation behind this research came from the need to quickly create and modify three dimensional environments. This need was further identified for a broader area of computer aided design. The duality of 3D modeling and virtual reality warranted this research.

From a historical point of view, many CAD systems were designed for 2D or 2.5D applications, and 3D capabilities were added on later. As a result, 3D related tasks were made to fit into the user interface designed for 2D tasks, giving little consideration to their unique characteristics and requirements. So it would be interesting to design an interface style that puts 3D quick prototyping at priority. In a sense, this is what this research is all about — to explore an old field from a new perspective, with new tools in hand.

Chapter 3

Related Work

As indicated in the previous chapter, computer-aided design (CAD) systems are being used mainly in the later stages of the design process, due to the long turnaround time of creating a model. In order to extend CAD systems into the early and creative stage, i.e. sketching, they have to provide easy and intuitive 3D interactions. Many attempts have been made to achieve this goal. We will give a brief survey in this chapter.

For technological reasons, 2D input devices were first developed and used for 3D direct manipulation. 3D input devices are relatively new and have just started being used. We will first review the important work using 2D pointing devices (such as a mouse or a tablet) in performing 3D interactions, then move on to work with 3D input devices.

3.1 3D interaction techniques based on 2D pointing devices

The difficulty in manipulating a 3D object with a 2D pointing device is obvious. There is a disparity in the number of degrees of freedom (DOFs). In order to fully determine the spatial state of a rigid body in space, one needs six parameters, three for position and three for orientation. If in addition to translation and rotation, the user also wants

scaling, then there are another three DOFs. However, a 2D pointing device only has two DOFs. The simplest way of supplying direct control over all these parameters is to present them separately, such as using one graphical potentiometer for each translation and scale, and using one graphical dial for each orientation. However, this type of control does not preserve the kinaesthetic correspondence between the 2D cursor movement and the actual 3D effect of the operation. A mental translation is required to coordinate the hand movement and the intended manipulation, thus placing a cognitive load on the user.

Several types of interaction techniques have been developed to add certain levels of kinaesthetic correspondence. The first type of technique is what we term *task partitioning*. Since there are only two DOFs available at any given moment in time, one cannot have simultaneous access to all the six DOFs. Instead, several mappings are designed to cover all the DOFs so that the user can access one or two of them at a time. The second type of technique can be called an *auxiliary object*. Instead of solely relying on the pointing device to supply all the DOFs, additional objects are used to assist the specification of parameters. This is analogous to the use of construction lines in geometric drawing.

In general, the task partitioning approach has the drawback that the user has to explicitly switch among different operation modes, because the task of selecting different mappings is extrinsic to the cursor control. However, it does not require existing objects, so it can be used for creating objects. For the auxiliary object approach, the task of specifying a mapping can be made intrinsic to the cursor control and thus becomes more intuitive. However, since it relies on existing objects for navigation, it is usually unsuitable for creating objects from scratch. It also tends to be less flexible, due to the reliance on existing objects.

3.1.1 Conventional approaches

For most commercial CAD systems, direct manipulations of 3D objects take the form of task partitioning.

There are two ways that a user can specify a 2D subspace for direct manipulation with the mouse or tablet. First, multiple views can be opened simultaneously, each corresponding to a particular mapping. For example, three orthogonal projections (e.g. the top view, front view, and right view) are opened for manipulation on the three major axes planes. An additional pictorial (perspective) view is often kept to aid the designer in correlating points among the different views. This approach uses the subdivision of screen space to provide easy switching among several mappings. It has the advantage that several mappings are easily accessible. The disadvantage is that each view takes a portion of the screen, causing real estate contention. As a result, frequent pannings and zoomings are often needed. Another disadvantage is the burden of keeping mental correlation among the different views. This problem is particularly acute when (a) the views have different zooming factors, and (b) the views focus on different parts of the object. Imagine looking at an object in a box through several holes, each fitted with a different magnifying glass, and showing different part of the object. In fact, an automatic mechanism of keeping all the views at the same zoom factor and focusing on the same point would help a lot. Unfortunately, most commercial packages do not have such a mechanism.

The second way of specifying 2D subspace for manipulations is by explicitly defining the mapping (sometimes called a user coordinate system, or UCS), and referring to them by names. For example, one can create a 3D box and use the edges on each face to define a corresponding mapping, and name them 'front', 'back', 'left', 'right', 'top', and 'bottom'. In the subsequent operations, the designer switches among these predefined mappings by supplying their names to a certain command. This approach allow the designer to access multiple mappings within a single view. In this way, the designer no longer needs to switch views to change mappings. The burden of mental correlation among different views is also eliminated (although correlating the current view with a 3D mental model is still required). The disadvantage is that the designer has to remember the name of each mapping, as well as the nature of the current mapping (otherwise surprises will occur).

In most CAD systems, both methods can be used in combination. However, they are still far from intuitive and flexible.

Most CAD systems also fail to address the issue of 3D rotation. Many systems only support rotations around a particular point on the current 2D subspace. This means that in order to specify a general rotation in 3D, the designer would have to establish a mapping whose 2D subspace is perpendicular to the intended axis of rotation. This, in turn, often requires the construction of auxiliary lines, making the operation slow and tedious.

3.1.2 Recent advances

Many researchers have been trying to provide more intuitive and flexible ways of using 2D input devices for 3D direct manipulations. We review some of the important advances in this field.

Bier's skitter and jack

Eric Bier designed the *skitter* and *jack* techniques for scene composition (i.e. the precise placement of shapes relative to each other) [6]. A jack represents a Cartesian coordinate frame used for specifying a transformation (such as a translation or rotation). A skitter is a three dimensional cursor used for placing jacks. This interaction technique mainly addresses the control of the 3D cursor by a 2D pointing device.

The two key ideas for this interaction technique are the use of ray casting in locating a 3D point and a set of rules for determining the orientation of a jack. To move the skitter around on the surfaces of objects, the user moves a 2D cursor around on the screen pointing at a projection of the 3D scene. The skitter appears on the surface of one of the objects visible in the pixel indicated by the cursor, that is, at the intersection of a ray fired from the view point through the 2D cursor to the surface of an existing object. Figure 2 shows a skitter moves on the front and back surfaces of an extrusion. In order to determine the orientation of the skitter, the surface normal at the intersection point is calculated to give the Z direction, and a rule is devised

to supply the remaining degree of freedom by using the centroid of the object. The skitter can also be placed at the centroid of an object. A jack can be placed wherever a skitter can rest.

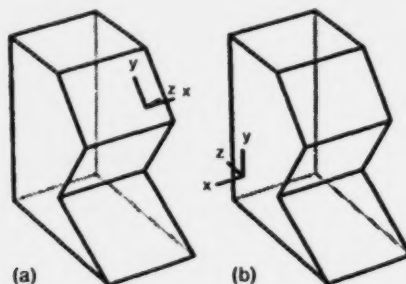


Figure 2: A Skitter moves on the surface of an extrusion. The two dimensional cursor is not shown. (a) on front surface. (b) on back surface.

The jacks placed on the surface or at the centroid of an object act as handles in specifying transformations. For example, the user can align two objects by placing one jack on each of them and then making the two jacks coincident. In figure 3, two jacks are placed on the body and handle of a mug, and the two parts are then aligned. The axes on a jack also serve to constrain translational motions of an object to be either along one of the axes or on one of the axes planes. This limitation on the number of degrees of freedom patches the gap between what the task demands and what the device supplies.

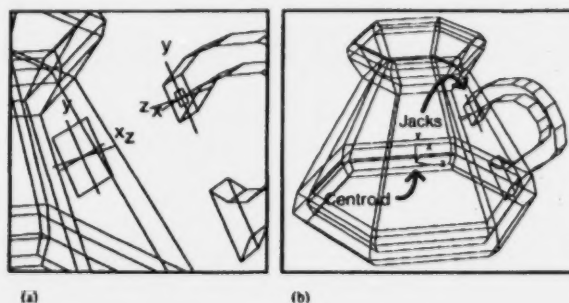


Figure 3: Aligning two objects with jacks. (a) two jacks are placed on the body and handle of a mug. (b) after aligning the two handles.

This technique is a good example of the task partitioning approach and the aux-

iliary object approach.

There are several limitations.

1. It relies on existing objects for the location and orientation of the skitter. So it is unsuitable for object creation or reshaping.
2. It uses surface normals to determine the orientation of the skitter. This imposes a restriction on the type of objects. It is difficult to handle geometric entities of low dimension (such as lines or points).
3. The support for specifying an arbitrary 3D rotation is very limited.
4. The task partitioning requires the user to switch among several operation modes, and reduces the transparency of the interface.

Bier's 3D snap dragging

After developing the skitter and jack techniques, Bier extended his work into snap-dragging in 3D [5]. The idea is to combine three interactive techniques that work well together: gravity, alignment objects and interactive transformations. While all these three techniques had been previously explored in various systems, several improvements were made in combining them together. First, different snap modes were merged to save switching time. Second, a single menu selection could result in the creation of many alignment objects, thus reducing mouse actions. Third, all of the smooth-motion transformations follow the skitter and can be given precise end conditions using gravity. These improvements achieved economy of user interface by taking advantage of real-time feedback and by performing precomputations. In order to combat the lack of direct 3D input, it resorted to ruler and compass metaphor for 3D construction. A special *anchor* object is introduced to help establish reference frames for manipulations. An anchor is the encapsulation of a coordinate system. By attaching an anchor to a special point (such as a vertex, an edge, or the center of an object), the user establishes a coordinate system for subsequent manipulations. For example, it can be used as the center of rotations, or supply an axis of rotation.

The main purpose of the 3D snap-dragging technique is to achieve interface economy, that is, to reduce the size of the interface (in terms of number of commands and number of menu entries). Since it uses ruler and compass metaphor for 3D construction, it is designed more towards drafting than towards sketching. Since every object is built from points to edges to faces, the technique has great flexibility. However, such an approach also slows down the construction process, as even a simple object like a box has to be built bottom-up. The major advantage of the 3D snap dragging technique lies in its reduction of the interface size.

Nielson's triad mouse

Gregory Nielson and Dan Olsen Jr. designed the "triad mouse" technique for direct manipulation of 3D objects using 2D locator devices [30]. The main idea is to map the 2D movement of the mouse to only one of the three axes of the space at any given moment. Clearly, this is a task partitioning technique.

The triad mouse is manipulated in 3D by repeatedly sampling the 2D cursor and using the 2D direction of its movement since the last sampling as the application's basis for a movement in 3D. In order to determine the mapping, the axis triad of 3D coordinate space is projected onto the viewing plane, yielding three projected axis vectors (see figure 4). After comparing the direction of the 2D cursor movement (i.e. (D_x, D_y)) with those three axis vectors, the axis whose projected vector is closest to the cursor movement direction is selected to be the direction of 3D movement. This solves the 3D location problem.

When the projected axis vectors are well separated, this technique offers a nice way of locating 3D points. However, it also has its drawbacks. First, the need to obtain a non-degenerate projection of the axis triad imposes a restriction on the viewing angle, namely, the line of sight can not be near parallel to any of the three axes. Second, the two available DOFs of the input device are not fully utilized, as the 3D movement is always restricted to be along one of the axes. In other words, one input DOF is used only for choosing among three options (a task which probably

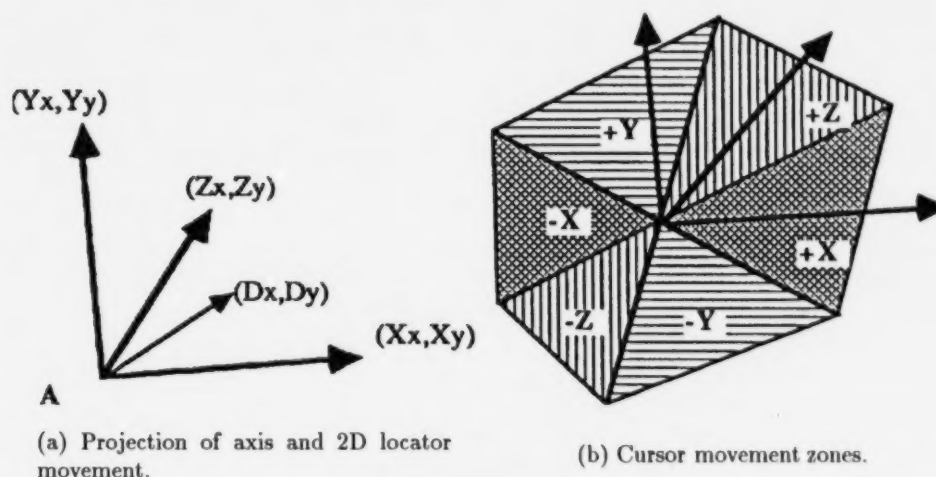


Figure 4: Triad mouse technique.

could be better handled through other input channels such as button presses). Third, the support for specifying orientation is inadequate. Performing a rotation requires a three-step process. One first selects two 3D points to define the axis of rotation. Then a pair of 3D points are selected to determine the amount of rotation. After that, the rotation command is issued to carry out the actual rotation. As such, it is hardly direct manipulation, since there is no kinesthetic correspondence.

Van Emmerik's grips technique

Maarten van Emmerik developed a technique for 3D manipulation with a 2D input device similar to that of the triad mouse [43]. Instead of just focusing on 3D translations, he tackled the problem of translation, rotation, and scaling at the same time with a more sophisticated task partitioning technique.

The main idea of this technique is the use of grips on the 3D cursor to assist task partitioning. Hereafter, we will refer to this technique as the grips technique. The 3D cursor is adorned with seven grips (handles), one at the origin, and one along the positive and negative direction of each axis at a fixed distance from the origin (see

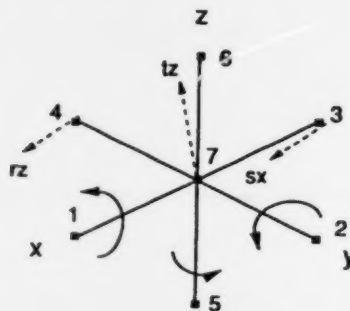


Figure 5: Grips on a 3D cursor.

figure 5). The user can select and drag one grip at a time with the 2D cursor. If the user drags the grip at the origin, the action is interpreted as translation along the axis whose projected direction is closest to be parallel to the direction of the dragging, just as in the triad mouse approach described above. For example, dragging the origin (grip 7) upwards (as shown in figure 5 by a dashed line) causes a translation along the Z axis ("tz" meaning translation along Z axis). If the user drags one of the non-origin handles along its corresponding axis, the action is interpreted as a scaling along that axis. In figure 5, a dragging of the grip on the X axis is shown in dashed line, with the label "sx" denoting a scaling along X axis. If the dragging direction is not along the axis, it is interpreted as a rotation around one of the other two axes whose projected direction is closest to be perpendicular to the dragging. In figure 5, a dragging of the grip 4 is interpreted as a rotation around Z axis. In this way, all three basic transformations (translation, rotation, and scaling) can be performed by direct manipulation.

Just as in the case of the triad mouse, the grips technique is also subject to restrictions on projections. In order for grips to be operable, they have to be well separated after projection. This imposes a restriction on viewing angle. Since the nature of the transformation depends on the instantaneous movement direction of the 2D cursor, the user may cause unintended slight changes to the state of the object without noticing it. The use of grips may also create a visual clutter problem as each 3D cursor takes up a relatively large amount of display area.

Kalay's worldview system

Yehuda Kalay's Worldview system [23] is one of the attempts made at using 2D interaction techniques for 3D interactive geometric modeling. It tried to combine the advantages of a drafting tool and those of a solid modeler. The Worldview system used multiple views to facilitate 3D spatial perception and manipulation. In order to select a 3D point, the user first picks a point in one view, effectively firing a ray from the viewing point through the 2D point. The ray is visualized in the other views as lines. The user then switches to another view and picks a point on the line representing the first ray. Figure 6 illustrates the idea of locating a 3D point by multiple hit-testing through nonparallel views.

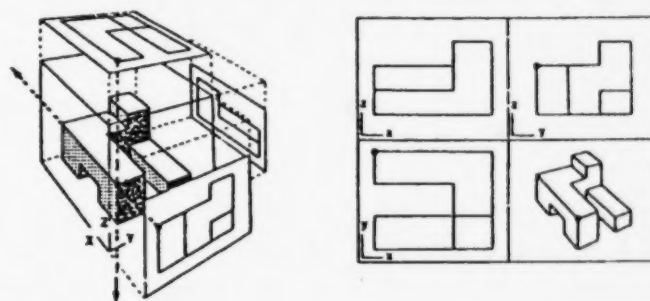


Figure 6: Multiple hit-testing through nonparallel views.

This technique can be considered as a combination of task partitioning and auxiliary object (the first ray is used as an auxiliary object). Since this technique can only be used for picking a 3D position along a ray, it is not very useful for general 3D manipulations.

Sittas' technique

E. Sittas designed a mouse-based system for 3D wireframe modeling [39]. It is another example of the auxiliary object technique.

This interaction technique uses three orthogonal projection planes to facilitate establishing 3D positions. A 3D point is defined by selecting two points on two different projection planes. A finite size construction plane can be used to construct

curves and other shapes on a plane not parallel to any axis planes. The system was intended for a sketching tool for 3D wireframe models, and does not support general 3D rotations. This technique is very similar to the one used in the worldview system discussed above.

Three dimensional widgets

Andries van Dam and his colleagues at Brown University developed a collection of so-called three-dimensional widgets for direct interaction with 3D objects [12]. This is another good example of the task partitioning concept.

The main idea is to use widgets (in the form of grips) to encapsulate geometry and behavior specification. For example, a rack consists of sticks (representing the reference coordinate system) and handles (representing low DOF operations such as translation along an axis) that can be used to manipulate an object. An improvement over the grips technique mentioned above is that the function of selecting the operation mode is transferred to mouse buttons, making accidental mode switches less likely. The distribution of functions over different grips also makes each task more constrained, and enables more functions to be incorporated.

One drawback of this technique is that it requires a lot of display area for visualizing the widgets. When there are many objects in the scene, other objects may occlude the grips, making them inaccessible.

Shoemake's Arcball

Shoemake designed the Arcball technique for direct manipulation of 3D rotations by a 2D input device [38]. The Arcball is based on the use of quaternions in specifying 3D rotations. The user is presented with a virtual sphere displayed as a circular disc. The user first picks a point on the surface of this imaginary sphere and drags it around (see figure 7). A great arc is drawn from the initial point to the current point. This great arc defines, by right hand rule, a vector $\mathbf{a}$ that is perpendicular to the plane on which the arc lies. This great arc also determines an angle β subtended at the

center. The output rotation is taken to be around the vector $\mathbf{a}$ with an angle 2β (i.e. double the original angle β). Note, in figure 7, the angle β is marked as $\theta/2$. The manipulation is also memoryless in that the result is fully determined by the starting point and end point, and is independent of intermediate points of the cursor. For these two reasons, the Arcball probably is the best technique for specifying a general 3D rotation with a 2D pointing device.

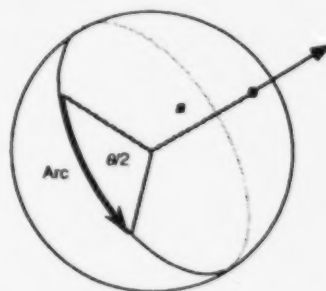


Figure 7: Arc interpretation.

One drawback of the Arcball technique is that it needs a relatively large amount of display space in order to achieve a reasonable level of accuracy. It often makes it impossible to place the Arcball at the same position as the object being oriented, and thus reduces the kinaesthetic correspondence. Another drawback is that not all rotations are accessible once a starting point is set, although this is no major problem as the planning of starting point is intuitive. Yet another drawback is that it does not lend itself to translation control. This demands a separation between orientation control and position control, again undermining kinaesthetic correspondence.

3.1.3 Section summary

While many of these techniques are useful for certain operations, they all suffer from the following limitations.

First, simultaneous translations along all three directions are not possible, due to the 2D nature of the device. This is one of the fundamental obstacles barring 2D locators from being useful for 3D quick prototyping. When a designer wants to

create a box-shaped object, he would wish to do it in a single stroke, as in the case of creating a rectangle with a 2D drawing tool. If a system forces the designer to break such a task into sub-tasks (such as creating a rectangle, and then adding the third dimension), the designer will have to think in terms of subspaces, rather than focus on the 3D nature of the object. In other words, the level of abstraction will be lowered, and the result is a low interaction efficiency.

When the task partitioning concept is employed, the user is burdened with the need to switch among different modes. Changing the mapping from 2D cursor movement to the corresponding 3D cursor movement reduces the kinaesthetic correspondence. For example, if the user has to switch among different views, then he has to correlate different points from different views to establish a mental model of the 3D shape. When auxiliary objects are used, the flexibility of manipulation is often limited. This often keeps the interaction technique from being useful for prototyping purposes. The use of auxiliary objects can also cause visual clutter.

In general, 3D rotations are not effectively supported. With the exception of the Arcball, most techniques limit the rotations to be around one of the major axes. To effect a general 3D rotation, three consecutive rotations have to be performed. The fact that rotations are not commutative makes the situation worse. As a result, it is difficult to orient a 3D object in space. The Arcball does allow for continuous direct manipulation of both the rotation axis and the rotation angle, and is probably the best technique for specifying 3D rotations by 2D devices.

In summary, 2D pointing devices are not very suitable for 3D interaction, in particular 3D prototyping.

3.2 High-DOF interaction techniques

Since 2D interaction techniques are inadequate for 3D direct manipulations, many experiments have been carried out in using high degrees of freedom interaction techniques to support 3D modeling. With the development of 3D tracking technologies, it

became possible to input real-time 3D information into a computer. Such 3D tracking devices can be used as hand controlled input devices. They can also be used to dynamically monitor head position and create the illusion of 3D immersion (with real time graphics output). We review 3D interaction techniques for both the immersive style and the non-immersive style.

3.2.1 Immersive style

There are two major research works using the immersive style.

Clark's pioneering work

In mid 1970's, James H. Clark built an experimental system for designing surfaces in 3D [11]. It was probably the first 3D modeling system ever built using a direct manipulation interface.

The main features of the system are:

1. the surfaces were rendered as isoparametric line drawings on a head-mounted display,
2. the surfaces were designed with the help of a 3D "wand", which allowed 3D movements of the points controlling the shape of the surfaces, and
3. all the interactions with the surfaces were in real-time.

In designing a surface, the user would first type commands on a teletype to select a stored surface-description file. The system would then display the surface on the head-mounted display as a stereoscopic scene. The user could change the shape of the surface by grasping control points with the wand. The wand was drawn as a small cube. The user could position the wand so that the desired control point was inside the cube, then he could press one of the wand buttons to grab the control point. The position of the control point would follow the position of the wand as long as the button was held down. The surface patches that were affected by the control

point were modified as the point was being moved. Since the display was updated in real-time, the user had the sensation of stretching a flexible sheet of material. The surface would retain its original derivative continuity because of the mathematical properties of the underlying B-spline representation.

The system had great advantages over the more common 2D techniques of viewing and designing curves and surfaces, in the ease with which it could be used to interact with 3D surfaces.

This set up has two limitations. First, fatigue. Since one has to assume a standing position with arms stretched out, he or she will soon experience physical fatigue. This makes it unfit for being used for an extended period of time, such as several hours a day. Second, precision. Three factors affects the precision of manipulation with this style:

- the user's arm is unsupported, giving way to unsteadiness and vibrations,
- the wand is controlled by wrist and arm motion, which are less precise than finger motion.

The lack of textual input during the modeling process didn't help either.

Despite these limitations, this pioneering work broke new ground in the use of 3D direct manipulation for geometric modeling. It first showed the power of 3D interactions in geometric modeling. Unfortunately, most technologies involved were not mature at that time and more than a decade passed before people picked up this thread again.

3DM

At roughly the same time this dissertation work started, Butterworth et al were building the 3DM system at the University of North Carolina [9]. The system used a head-mounted display to place the designer in the 3D modeling space. The input device was a six degrees of freedom tracker with two buttons on it. The system was modeled after the popular 2D drawing tool MacDraw. A user would use the hand-held

tracker to select functions from a toolbox (visualized as a flat panel in 3D space), and to create or manipulate objects in 3D space. The system demonstrated an easy and intuitive way of modeling 3D objects. For example, a user would first pick a primitive shape from the toolbox, then created an instance of shape using the 3D cursor. He could also move or reshape an object with the 3D cursor.

Compared with Clark's system, hardware technologies have improved significantly. Software technologies have also advanced a lot, as evident from the use of graphical menus and such. However, the intrinsic problems with an immersive style interface remains. On the fatigue side, the user still assumes a standing posture with an arm stretched out. Since the 3D sensor is mounted inside a hollowed out billiard ball, the user can now use some finger motion to control it, on top of wrist and arm motions. However, the unsteadiness problem with an unsupported arm remains. The lack of access to keyboard and mouse makes most traditional interaction techniques unavailable to the user.

WIM

Recently, Stoakley et al developed a user interface technique called World in Miniature or WIM [41]. It augments an immersive head tracked display with a hand-held miniature copy of the virtual environment. In addition to the first-person perspective offered by a virtual reality system, a WIM offers a second dynamic viewport onto the virtual environment. Objects may be directly manipulated either through the immersive viewport or through the 3D viewport offered by the WIM. Providing multiple perspectives in a 3D environment is a very interesting idea, as it has the potential of resolving the contention between depth perception and visibility.

3.2.2 Non-immersive style

Several researchers have experimented with non-immersive style 3D interaction techniques.

Schmandt's system

Christopher Schmandt built an experimental system at MIT [37]. The "interactive stereoscopic computer graphic workspace" used PLZT shutter glasses to provide a binocular stereoscopic display for the user. By using a half-silvered mirror, the graphics display was overlaid on a real world scene. A Polhemus six degree-of-freedom tracker was used to dynamically monitor the position and orientation of a hand-held wand. Figure 8 shows the overall design.

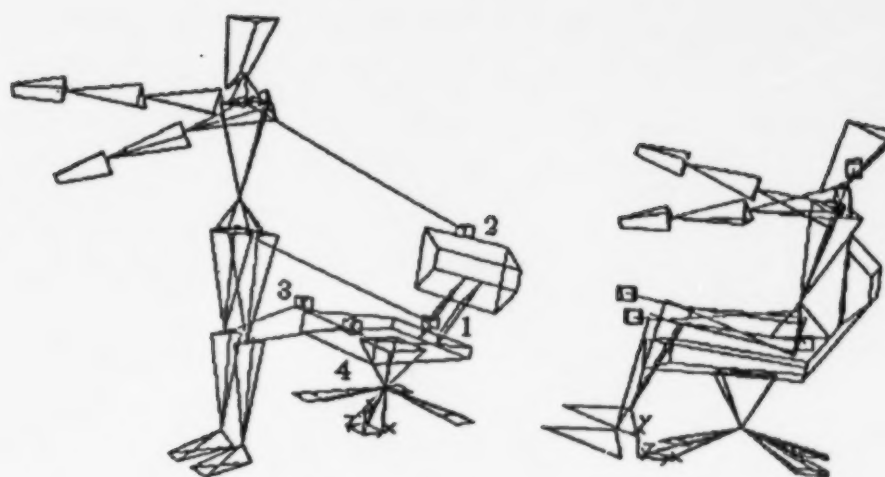


Figure 8: A cutaway view of Schmandt's workstation system.

Badler's technique

Norman Badler et al designed a 3D positioning technique using a Polhemus Isotrak [4]. The six degrees of freedom input device was used to select predefined 3D points on an articulated figure, and to specify their goal position with respect to the environment. Figure 9 shows the specification of goals for a sitting position, and the final result.

The experiments showed that the use of a multi-dimensional input device could make positioning tasks less tedious than manually adjusting coordinate values. They



(a) Specifying goals for a sitting position.

(b) Solution for sitting goals.

Figure 9: Using 3D input device to position an articulated figure.

also acknowledged the lack of depth cues on 2D displays for 3D direct manipulation tasks.

3-Draw

Emanuel Sachs et al built the 3-Draw system for interactive 3D shape design [34]. This system was based on a pair of six degrees of freedom input devices, one held in each hand. This enabled the designer to position and orient objects relative to each other in a displayed virtual world with the same ease as manipulating two hand-held objects. The simultaneous use of two sensors took advantage of people's innate ability — knowing precisely where their hands are relative to each other. The kinesthetic feedback from the hands simplified shape design by providing natural ways to manipulate a 3D drawing/editing tool and the object being created. In using 3-Draw, the designer would hold a sensor (in the form of a palette) in his left hand. This corresponded to a moving reference frame. In his right hand the designer would hold a stylus that corresponds to a drawing/editing tool. The designer could create a

3D curve by drawing with the stylus. He could then use the stylus to edit the curve. Curves could also be connected to form 3D wireframe models.

The 3-Draw system demonstrated that developing a 3D model by drawing directly in 3D is natural and quick. It showed that strong kinaesthetic feedback enables a designer to interact with 3D objects rapidly and naturally.

3.2.3 Section summary

These systems all have a single view, as the ability to intuitively change the viewing set up eliminates the need to have multiple simultaneous views. This also removes the two cognitive burdens: (a) correlating points from different views, and (b) switching between different mappings from hand movements to 3D motions. These are the main advantage of using input devices with high degrees of freedom.

The main reasons that none of the above mentioned approaches has evolved into a usable system include:

1. failure to recognize the importance of traditional keyboard and mouse interaction techniques,
2. failure to provide an ergonomic environment,
3. failure to supply sufficient precision, as required by most engineering disciplines.

3.3 Chapter summary

From this brief survey of relevant work, we can make the following two observations:

First, traditional interactions techniques are tethered by the 2D input device, and thus are inadequate for efficient 3D modeling.

Second, early attempts at employing 3D direct manipulation overlooked the importance of traditional interactions and other issues (such as precision, and ergonomics).

Recently, Hinckley et al did an excellent survey on design issues in spatial input [19], which provides a more in-depth discussion on human perception and ergonomics

concerns.

In the next chapter, we will propose a new style for 3D solid modeling that can cope with these problems.

Chapter 4

Interaction Style — The Design Philosophy

In the previous chapters, we discussed the motivation for this research and did a brief survey of related works.

Now we will discuss the interaction style for 3D solid modeling, based on the two observations we made so far. We give an overview of the style here together with the philosophy behind major design issues. More detailed discussion will follow in the next chapter.

First of all, as we have seen in the discussion of related work, there are two pitfalls in design philosophy: conservatism and radicalism. By conservatism we mean the clinging to traditional I/O devices and the associated interaction techniques. By radicalism we mean the total displacement of traditional input devices and their associated interaction techniques by new 3D I/O devices. In designing the new style, we try to avoid both extremes by seeking a congruence of the two types of I/O devices and interaction techniques.

The design of user interfaces is by nature multi-disciplinary and multi-faceted. Recognizing this fact, we try to strike a balance between the demands from various aspects of the problem. The new style can be viewed as the result of several design constraints coupled with the capabilities and limitations of current technology. The

major design constraints come from ergonomics, engineering requirements, and prevailing practices. The current technology includes real time 3D tracking and modern graphics hardware.

Clearly, the task of 3D solid modeling demands a 3D interface — one that preserves kinaesthetic correspondence. As we have discussed in the previous chapters, this cannot be achieved by using 2D pointing devices. Instead, 3D input devices have to be incorporated in order to unshackle the communication channel between the user and the modeling system. While such an idea seems largely foreign to the CAD user and developer community at large, it is a necessary change to meet the challenge of 3D modeling.

Nevertheless, this does not imply that traditional input devices and interaction techniques should be thrown out. Previous attempts at introducing 3D input devices into 3D modeling suffered from several problems, many of them have their roots in a type of interface radicalism. First, it assumes that traditional devices can be easily thrown out. However, throwing out devices also throws out the skills that people have painfully developed and learned over the years, which results in users' cultural resistance to the new interface. Second, the radical view assumes that a 3D device can replace all existing devices. However, this is not the case. The fact that a 2D device is inadequate for 3D manipulation does not mean it is not suitable for any interaction. For example, when it comes to navigation through 2D hierarchical menus, a mouse is still the best input device. For another example, textual input is critical for annotation and necessary for precision. To this end the function of a keyboard can hardly be replaced by any type of 3D tracker. A good example comes from the 3DM system at UNC. Since it uses an immersive head-mounted display for output, the user totally loses access to a keyboard. As a result, any model a user creates must first be saved into a file with a fixed name, then renamed afterwards. There are also a lot of operations that are 2D or 1D in nature, and thus are well suited for a 2D device. Third, the radical view assumes that the availability of 3D manipulation would override other concerns such as ergonomics. However, lack of

ergonomic consideration (such as fatigue factor) rendered most of these early systems infeasible as solutions to engineering applications.

Another common problem with these previous attempts is the simple adoption of 2D interaction techniques into 3D manipulation. This is mainly due to a lack of understanding about the unique characteristics of 3D devices and 3D manipulations.

In designing our interaction style, we try to avoid this type of interface radicalism and add 3D devices to the existing set of devices rather than displace them. We also try to adapt interaction techniques to better suit 3D devices.

From these observations, we design the style to provide:

- 3D kinaesthetic correspondence,
- easy access to all the input devices and their associated interaction techniques,
- 3D interaction techniques that take into consideration the unique characteristics of 3D modeling tasks and those of the new input devices,
- support for quick prototyping and the smooth transition into later design stages.

4.1 Overall set up

As we discussed in the previous chapters, one major reason that previous attempts to bring 3D direct manipulations into modeling failed is the lack of ergonomic considerations for the user. Since an engineering designer usually spends several hours a day on his designs, the ergonomics of the system must not be neglected. There are two aspects to this issue: physical constraints and fatigue.

For ergonomics reasons, a system should impose as few as possible physical constraints to the user. From this point of view, tethering devices such as head-mounted displays and instrumented gloves should be avoided as much as possible. If a designer has to don a diving mask and put on a glove, both chained with cables, in order to use a system, then chances are that he would prefer a less intrusive setting. The confinement of these devices will interfere with his other daily activities such as looking

at a normal computer screen, reading paper-based reference materials, locating and talking to his colleagues, answering the phone, answering the door, reaching for a coffee mug, to name a few. He would rather use a system that blends better into all the rest of his activities. Such a system must be easy to engage and disengage.

A related but slightly different issue is fatigue. If a system requires the user to assume a tiring posture, then it will easily result in physical fatigue. For example, the use of an instrumented gloves usually requires the hand to stretch out, and will cause arm fatigue if used for an extended period of time. A head-mounted display usually adds a few pounds of weight to the head, and will cause neck fatigue. This problem is often exacerbated by the lack of proper counterbalance.

Based on these observations, we selected a non-immersive interaction style as the basis of our style. The overall setting of the style is shown in figure 10. This style is sometimes referred to as Fish Tank Virtual Reality [44].

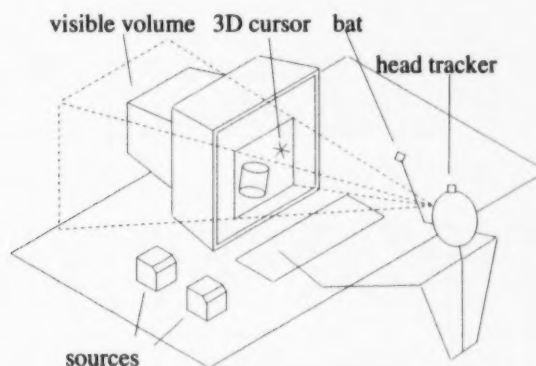


Figure 10: Overall setting of the style.

The designer holds a 3D sensor in his hand, which serves as the main input device, and his elbow rests on the desktop. This provides the freedom of 3D manipulations while keeping physical constraint and fatigue at a very low level. Since the 3D sensor can be picked up and put down as easily as a pen or pencil, the effort required for engaging and disengaging is minimal. This provides a flexible yet congenial interface style for the designer.

Since the designer's elbow rests on the desktop, the precision of hand position

control is higher than an unsupported out-stretched hand, as the case with 3-Draw, 3DM and other glove based system. This precision gain is needed as we have observed in the previous chapter.

This non-immersive setting also allows the designer to access all the traditional input devices such as the mouse and the keyboard, as well as other things such as a notepad, a pencil, a telephone or a coffee mug. The access to the keyboard and the mouse is very important. It not only makes all the traditional software available to the designer, but also provides the designer with the ability to issue textual input.

The access to software tools outside the modeling system provides a smooth blending of the sketching stage with later stages. For example, the designer may perform a finite element analysis using a command line driven tool, and have the output displayed in a graphical window, controlled by the mouse. Such flexibility would not be available, if the designer was wearing a head-mounted display.

The access to textual input is also very important to engineering design, as it is the ultimate way of specifying precise values (such as dimensions). It compensates the inherent inaccuracy of direct manipulation.

The need for accuracy is ever present in engineering design. In early stages, one needs precision to ensure that the prototype matches certain design constraints, such as a standard chassis or fixture. It saves a lot of refinement effort which would otherwise be required later on. The access to the keyboard provides the most effective solution for precision.

As the design process continues, the need for precision grows. With our style, the designer can simply phase out some of the direct manipulations and phase in more accurate methods. In this way, there will be no sudden change in the design style, and a smooth transition from sketching to detailing can be achieved.

In summary, this style provides a 3D interface without incurring many disadvantages suffered by early attempts.

4.2 3D interaction techniques

A major part of the new design style is 3D interaction techniques. As we have noted before, 2D interactions are not adequate for 3D modeling tasks. So we have to develop new techniques that can break the 2D bonds. However, since 3D input technologies are relatively new, we do not yet have a good understanding of many issues related to the use of 3D control. Many early attempts at making use of 3D input were simple imitations of 2D techniques and did not make full use of their potential. Also relatively little consideration was given to the unique characteristics of 3D manipulations. Sometimes they are even less effective than their 2D counterparts.

In developing the 3D interaction techniques for our new style, we keep in mind the fundamental difference between 2D and 3D manipulations, and explore novel and effective ways of utilizing the great potential of 3D inputs. We also keep in mind the specific requirements of engineering design, and try to tune the interaction techniques towards this special application area.

At the beginning of this research, we did not have much experience with 3D manipulations. As we will see in the next chapter, many early designs did not perform as well as we had wished. With the accumulation of experience, we identified problems, and gained insight into unique issues in 3D manipulations. Refinements to the techniques were carried and the cycle began again. The results so far are by no means the be-all and end-all of 3D manipulations, as are evident from their paths of evolution, but they do provide significant insight into the nature of 3D direct manipulation.

4.3 Other issues

As we discussed in the previous chapters, one problem associated with the 2D metaphor is the low level of abstraction in modeling. Since there are only 2D based notions such as line, circle, etc., the designer sees a 3D object as a collection of these entities. As a result, he often has to rely on construction lines and such to represent high level concepts such as the center of an object. In promoting the interaction into

3D, we can easily provide these features to the designer at the object level. However, this also demands that the basic modeling operations be based on objects rather than lower level entities. To achieve this goal, we provide the designer with constructive solid geometry (CSG) operations as the basic modeling operation. While CSG operations themselves are not within the scope of this research, their implementation did significantly increase the complexity of the prototype system and the amount of programming effort thereof. We consider such extra effort justifiable because it reveals the real merits of our style rather than leaving it only as a mock-up toy.

Once we lifted the level of abstraction into 3D objects, many new issues arise. For example, in order to help the user understand the 3D shapes, they should be rendered as solids for most of the operations. However, there are cases when the occlusion among solids becomes undesirable. So new visualization methods were developed to reveal the full details of spatial relationship among solid objects while maintaining their 3D solid appearance for visual interpretation.

The need for kinaesthetic correspondence also imposes a very strict constraint on system performance. None of the interaction techniques can afford to be computationally expensive. Otherwise, they will slow down the update rate of the display to break the illusion of real time response, and negate the whole purpose of providing 3D direct manipulations.

4.4 Chapter summary

In this chapter, we discuss the selection of a proper interaction style for 3D geometric modeling. We explain the path leading from our observations and design philosophy to the particular interaction style we chose. We criticize both the interface conservatism and interface radicalism embodied in early attempts, and aims at an optimal solution under all the design constraints.

Chapter 5

JDCAD — A Prototype Implementation

Starting with this chapter, we discuss a prototype implementation of our interface style. This system, called “JDCAD”, is built to study the merits of our interface style. First, we present a brief history of the system, as it has gone through several stages. Then we discuss low level issues such as 3D input and output, upon which other interaction techniques are built.

5.1 Brief history

The implementation of a prototype modeling system started in early 1992. Little is left from the first version as numerous revisions have been made, reflecting the insight gained into major issues in 3D manipulations.

The prototype system was first developed on an SGI Personal IRIS workstation, as it was the best hardware available to the author at that time. Later, two IRIS 4D/310 VGX machines were acquired by our graphics research laboratory, and the system was ported to the new machine, with major changes to take advantage of the better hardware and to improve software architecture. At the same time, colleagues in the lab got tired of referring to the anonymous system, and Chris Shaw finally

came up with the name "JDCAD", after the initials of the author's first name. Since the porting changed the software architecture of the system, the old version was left as a separate system. It was later revised by Shawn Halliday to incorporate animation functions, and renamed "JDCAD+". Meanwhile, implementation and testing continued on the VGX machines, and JDCAD became the "official" name for the system. When the graphics lab acquired its latest SGI machine, a RealityEngine, in mid 1994, JDCAD was ported to the new machine without major change.

For the sake of simplicity, we only use the term JDCAD in the following chapters, without referring to specific revisions, as we are only interested in new interaction techniques, not the complete functionality of any particular version. However, we will briefly mention JDCAD+ in chapter 9, when we discuss applications of our interface style.

5.2 3D input device

JDCAD uses a Polhemus Isotrak six degrees of freedom tracker as the main input device. To enhance control, we mount three Omron JP3101 mechanical keyswitches, which are momentary pushbuttons that have an audible and palpable click. This type of button-enhanced tracker is sometimes referred to as "bats" [45]. In the usual posture, the thumb can press the tail button (nearest the wire), and the dorsal button (the middle button). Shifting between the tail button and the dorsal button is a simple matter of rocking the thumb over the pivot of the button casings (figure 11 middle right). The index finger presses the nose button. The sensor is supported in the fingers by the middle finger and/or the third finger.

There are many issues involved in the selection of 3D input devices. We list the important ones, and show that the Isotrak has overall advantages over other type of hand controlled 3D input devices, such as instrumented gloves, mechanical tracker, or 3D mouse.

Ease of engaging and disengaging.

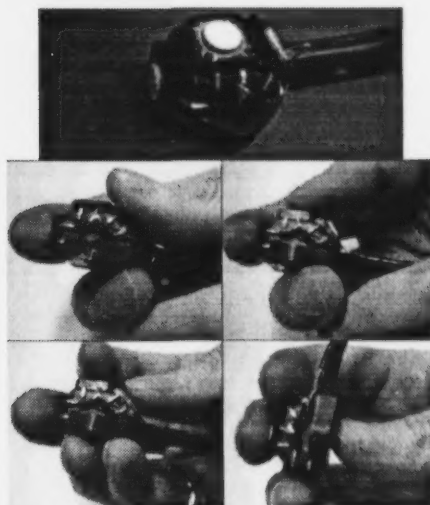


Figure 11: Top: The button-enhanced bat. Middle and bottom: Four postures for holding the bat.

Most 3D tracking devices currently available use cable for connection. This imposes a physical constraint on the user, which, depending on the length of cable and other configurations, may hinder or even forbid the user from performing certain activities, such as picking up a phone or a coffee mug. For this reason, it is desirable to use a device that permits quick engaging and disengaging. Picking up an Isotrak sensor from the desktop or putting it back down usually takes less than a second, while putting on a glove and taking it off usually takes five to ten seconds.

Physical surrogate.

It is desirable to have a physical surrogate in hand when manipulating a 3D virtual object. Imagine rotating an orange in hand. The finger tips alternate between holding the orange in place and moving across its surface. It is the tactile feedback that enables the proper control of force exerted at each finger. With the Isotrak sensor in hand, the user has the necessary tactile feedback for rotating a virtual object with similar ease as rotating a physical one in hand. For this reason, it is preferred over gloves.

Fatigue factor.

It is desirable to use a small and light-weighted input device when the user needs

to use it in 3D. For this reason, an Isotrak sensor is preferred over larger and heavier devices such as the 3D Mouse.

Flexibility of motion.

The flexibility of motion is closely related to the posture for holding a device, which, in turn, is determined by the size, shape, and weight of the device. Devices controlled by finger motions are more flexible than those controlled by wrist or arm motions. For example, the Isotrak sensor can be continuously rotated by finger tips for several turns until cable tension starts to take effect. The same flexibility does not exist with a 3D mouse (in which case, all the fingers are holding the device leaving only wrist and arm motions for control, and the accessible range of rotation is limited to within one full turn). For the same reason, most mechanical trackers are not suitable either.

Precision of control.

It is well known that different parts of the body has different precision of motion control. For example, finger tips have higher precision than palm and wrist. In this case, the small overall size of the Isotrak sensor makes it possible to be manipulated with finger tips, instead of wrist. This makes it preferable than larger devices, such as 3D mouse or the instrumented billiard ball used in 3DM.

Isotropic shape.

The relatively isotropic shape of the Isotrak sensor makes rotation control more uniform, compared with the stylus shaped sensor used in 3-Draw and the handle-shaped bat use by Ware. This is preferable when input of arbitrary rotation is needed.

Overall, a device like Isotrak is preferred over other type of 3D input devices for our purposes. Of course, a light weight wireless tracker would serve even better for its flexibility and unlimited range of rotation. However, there is no such device available yet.

5.3 3D display

To enable 3D direct manipulation, a system must provide a 3D environment where the interaction can take place. Since we do not yet have true 3D display techniques, we have to rely on 2D displays to visualize a 3D scene. In order to be compatible with existing interaction styles, we borrow the "window" concept employed by most workstations, and use a rectangular area of the screen as a window into a virtual 3D space (returning to the original meaning of the word *window*). All the interactions take place in a visible frustum defined by the user's view point and the rectangular area on the screen (cf. figure 10).

In order to supply the user with as much depth information as possible under performance constraints, JDCAD employs several visual cues to visualize 3D objects.

First, we render the objects as solids to provide occlusions among objects. This supplies the most important qualitative depth information of the scene [15]. Second, we render the surface of objects using Gouraud shading from an attenuated directional light at close distance. This lighting provides the user with surface orientation information and partial depth information.

In addition to these common depth cues, we also provide an optional kinetic depth cue by tracking the designer's head position [13, 37]. When sitting in front of the graphics workstation, the designer wears the sensor of the tracking device on his head. Through a series of transformations, the approximate eye position with respect to the display window is obtained and used as the center for the perspective projection (cf. figure 10). When the display is updated more than 10 frames per second, the designer has a strong spatial perception as if he/she is looking into a real 3D space in front of him.

It would also help to employ field sequential stereo display, but due to limited resources, we were unable to explore this option.

Similar experiments with kinetic depth effect were also made by other researchers[13, 37].

This setting provides a platform for 3D direct manipulation. It has several ad-

vantages over the immersive virtual reality style. First, the physical constraints on the user are minimal. There is no tethering device like a head-mounted display. The user can switch gaze from the CRT screen to any other places in practically no time, as there is no need to put on or to take off a head-mounted display. Second, the user does not suffer the eye fatigue and disorientation that would result from wearing a non-see-through head-mounted display. This is essential if the system is to be used for an extended period of time (e.g. several hours a day). Third, the user can use the entire surface of the graphics console, which has a much higher spatial resolution and color resolution than those of currently available head-mounted displays. This is especially important for fine manipulations. Fourth, the approach does not prevent the user from accessing conventional input devices such as keyboard, mouse, joystick, etc. This ensures a high degree of compatibility with existing interaction techniques. It also allows the user to access other software while using the modeling system. For example, he can easily spot any incoming electronic mails and read them in another window, without having to leave the modeling system.

5.4 3D input processing

We put the measurement data from the Isotrak through a series of transformations in order to achieve the correct kinaesthetic correspondence between hand movement and the 3D cursor motion, and to support various intuitive interaction techniques.

The tracking data from the Polhemus Isotrak is given in a device coordinate system, which is determined by the location and orientation of the Isotrak source. They usually have to be transformed into a useful reference frame first [36]. In JDCAD, we choose the graphics monitor as the base reference frame for coordinate transformations.

Several intermediate reference frames are necessary for achieving the correct kinaesthetic correspondence between hand movement and 3D cursor motion. To inspect an object and its relationship with neighboring objects, one may want to grab the

model with the bat and simultaneously rotate/translate it. Such operations require a reference frame whose origin is near the center of the visible volume. Zooming operations also have a similar requirement. For this purposes, we define an *apparent coordinate system*, which is centered at the design space. With this reference frame in place, the user can easily rotate the entire model in hand, and gain a better spatial perception from the motion parallax it creates. We observed that most users execute this operation frequently, indicating an inadequacy in spatial (depth) perception even with all the other depth cues present.

5.4.1 Filtering

The measurement data from the Isotrak usually contains noise and bears a certain amount of delay time [27]. The noise comes from internal imperfections of the Isotrak as well as external interference. It causes the 3D cursor to jitter around in position as well as orientation. This makes it difficult to precisely position or orient with the bat. The delay in measurement data mainly consists of the time spent on internal calculation within the Isotrak device and on transferring the data from the device to the computer. Such delay causes the cursor motion to lag behind actual hand movement, which in turn causes overshoot in manipulations. Due to these factors, we employ Kalman filtering to reduce the noise level and compensate for some of the lag [27]. From our experience with the Isotrak tracker, the lag is often more of a problem. It slows down the operation as the user tries to adjust for compensation.

5.4.2 Clutching

Since the Isotrak uses magnetic field for sensing, there is no easy way to quickly disengage and re-engage tracking. In other words, it is an absolute locator by nature. However, what we prefer to have is a relative locator, as it is often necessary to adjust, from time to time, the location mapping (offset) from hand position to 3D cursor position. For example, the system should allow the user to rest his or her

elbow on the desktop no matter where the 3D cursor is in the design space. This is very important from an ergonomics point of view.

To turn this absolute locator into a relative one, we introduce a clutching mechanism. The user can momentarily disengage the 3D cursor from hand movement by holding down a key (in this case, the "Esc" key), and re-engage it by releasing the key. This allows for very easy adjustment of the location mapping. A common scenario is that the user moves the cursor to the vicinity of an object to perform certain operations on it. If this action puts the forearm and hand in a somewhat awkward position, then he would disengage the cursor, place the hand in a comfortable position, and re-engage the cursor.

For the ease of certain operations (such as menu selection and object selection, which we will discuss later), the user may desire to have the 3D cursor at or near the center of the design space. To facilitate this, we build another clutching mechanism, called "homing". With a key press, the user centers the 3D cursor in the design space. This turns out to be a very frequently used operation, as the center part of the design space is preferred for most modeling operations.

With these two clutching mechanisms in place, the Isotrak can be used with comparable ease as a relative locator such as a mouse.

5.5 Chapter summary

Low level design issues affect all the interaction techniques built on top of them. We address these issues in light of ergonomic considerations, ease of use, and the unique characteristics of 3D manipulations, and devise supporting facilities accordingly. While far from a complete set of tools, they provide a reasonable base for most 3D interactions we will discuss later on.

Chapter 6

Menus, Selectors, and Dials

In this chapter, we will discuss the development of three basic interaction techniques, namely, menu selection, object selection, and dials.

In our style, since the designer usually holds the bat in one hand, and uses the other hand for keyboard interactions (see figure 10), it is usually inconvenient to use the mouse, as it takes time to switch hand from one device to the other. Many interaction techniques discussed here are designed to avoid hand switching by utilizing the bat to accomplish tasks usually assigned to a mouse.

6.1 Menu selection

Menu selection is widely used in interactive systems to assist the user in making a choice among several candidates. By showing all the candidates to the user, the system relieves the user of the burden of remembering all the choices. Also, menu selection by a pointing device transforms the selection task from an textual interaction into a spatial interaction. This transformation has several implications. First, by arranging the menu items in a meaningful layout, the system can provide the user with more information about the choices. For example, by putting related menu items into groups, the system shows their close relationship through spatial proximity. Second, since menu items are displayed entities, they can contain richer information

than a textual list. For example, menu items may contain pictures (icons), which are more descriptive than texts in many cases. Faded-out menu items can also be used to indicate unavailable choices depending on the current state. Third, the transformation from a keyboard task into a pointing task changes the precision requirement for hand/finger motion. In keyboard tasks, the spatial precision requirement is more or less fixed, as there is not much variation in keyboard size and layout. In pointing tasks, however, there can be a large variation in this aspect, depending on the effective target area and distance. For example, a well designed menu selection scheme can incorporate choice prediction, and display the most likely choice under the current cursor position, eliminating the need for motion in many cases. On the other hand, if menu items are too small and the cursor has to travel long distances to reach them, then the user will spend more time, as governed by Fitts' Law [10].

These observations are the guidelines for designing menu selection techniques with 3D pointing devices.

6.1.1 Daisy menu

The first 3D menu technique we designed is called a *daisy* menu [26]. A daisy menu is a 3D spherical arrangement of menu items controlled by the bat. The menu items are evenly distributed over the sphere, for example, on the vertices of a dodecahedron or icosahedron. When the designer presses down the key for a pop-up daisy, the daisy shows up around the 3D cursor, and its orientation follows that of the bat. In addition, a selection basket (a truncated cone in wireframe) is displayed, which is centered at the 3D cursor and always faces the designer. The designer may rotate the daisy so the desired item falls within the selection basket (the item is color highlighted when it is in the selection basket), and releases the key or button to confirm the selection and dissolves the pop-up daisy. Figure 12 shows a snap shot of a daisy for choosing among various kinds of primitives.

Unlike most 2D menus, the menu items move instead of the selector. This derives from the fact that we perceive 3D space from 2D images, as our retinas are 2D. This

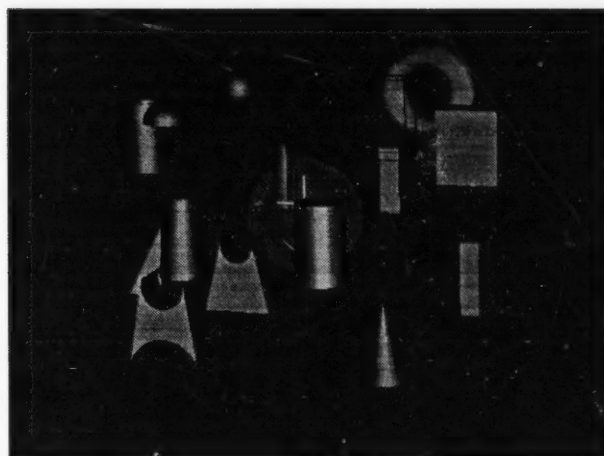


Figure 12: A snapshot of a pop-up daisy.

causes what we know as occlusions. If we make the menu items static, then we have to make sure that none of them occlude by other menu items on the selector. This essentially turns the menu selection technique back into a 2D one, wasting all the added DOFs. By moving the menu items with respect to a static selector, we avoid the static occlusion problem, and make good use of the space — as an extension of the 2D area.

The daisy worked quite well. It let the designer select menu items without switching the dominant hand between bat and mouse. The selection operation requires only finger movement, since the bat is only about $1 \times 1.5 \times 1.5$ cm in size and can easily be rotated using two fingers and thumb.

6.1.2 Ring menu

After using the daisy menu for a while, we discovered room for further improvement. First, we noticed that sometimes there could be a depth ambiguity among the items when the daisy is relatively static (as seen in figure 12). This is mainly due to the lack of adequate depth cues. Important depth cues for close objects, such as binocular parallax and difference in focal distances, are missing in our visualization scheme. Second, the cable extending from the sensor makes it anisotropic with respect to

rotation. Since the cable is extending along the negative X axis of the sensor, rotations around the X-axis are easier to execute than those around other directions. These two observations led to the design of the ring menu.

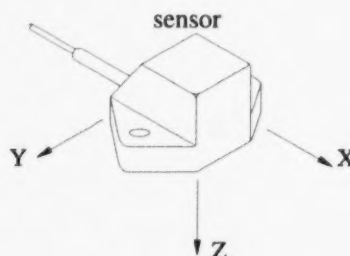


Figure 13: Directions associated with the Isotrak sensor.

In a ring menu, the items are arranged in a circular way (see figure 14). Two semi-transparent belts are used to eliminate depth ambiguity and signify selection. The gap on the front belt always faces the designer (just like the selection basket in the daisy), and the belts remain static with respect to the X-axis. The designer turns the items in the ring by rotating the bat around its X-axis. The item that falls into the gap is highlighted as the current selection. The designer may rotate the bat to tilt the entire ring in any direction, but only rotation around the X-axis turns the items with respect to the belts. In other words, menu selection is controlled only by the rotation around the X-axis of the bat. To better visualize the relationship between the 3D cursor and the menu items, spokes are drawn from the cursor to the menu items. This helps the user to associate the rotation of the bat with that of the menu items. The two belts are intentionally drawn in space, unattached to the cursor, to signify their invariant orientation.

The ring menu is superior to the daisy in its ease of use, as there is no depth ambiguity, and the rotation is easy to execute. This is one example where one has to recognize the unique characteristics of 3D space in order to produce satisfactory results.

Some people suggested the use of a 2D menu controlled by a 3D device, such as

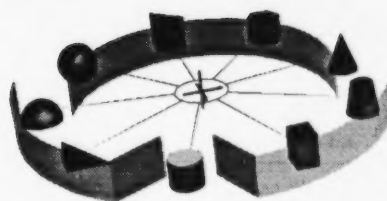


Figure 14: A snapshot of a ring menu.

a direct extension of the 2D pie menu. However, our experience shows that it tends to be distracting as it breaks the 3D perception of the space and reminds the user about the flat nature of the display.

One drawback of the ring menu is that it does not scale very well. We have not found a proper design for a hierarchical ring menu. This is also the case for the daisy menu.

6.1.3 Implementation notes

There are two issues to be considered in implementing these 3D menus.

First, orientation control. The idea behind these 3D menus is to make the menu items rotate with the bat while maintaining a relatively constant orientation for the selection indicator (the basket in the daisy menu and the belts in the ring menu). A proper selection of reference frame for the menu rendering can simplify the calculation of transformations. For the daisy menu, the terminal serves as a good reference frame, as the user is facing the terminal. The basket can be drawn facing “out” from the terminal. The orientation of the bat with respect to the terminal is then taken as the base for rendering the menu items. For the ring menu, preference of coordinate system is not so strong, since it only uses the rotation around the bat’s X-axis for selection. One only needs to transform the user’s head position (or a fixed point in front of the screen in case the head position is not being tracked) into the current

reference frame, and rotate the belts around X-axis to face it. Suppose the head position in the local coordinate system is (x, y, z) , then the rotation angle is given by $-\arctan(y/z)$.

Second, avoiding undesirable occlusion. In JDCAD, objects can be rendered as solids or wireframes or just bounding boxes. In normal cases, the user uses the solid rendering mode to get depth cue from occlusion. However, this may cause difficulty for 3D menus, as the menu may be occluded by other objects, either partially or completely. To avoid this problem, we temporarily turn the rendering mode to wireframe when the 3D menu is in use. This way, the menu is always clearly visible. The objects will return to solid appearance once the menu is dissolved.

6.2 Object selection

Object selection is another important task in interactive systems, as the user often operates on only one or a few of the objects at a time. In 2D modeling systems, the task of object selection can be executed in several manners. One way to select an object is positioning the cursor over a visible part of the object (e.g. a line) and clicking a button. Another way is using the cursor to define a rectangular area (window) and letting the system determine which objects intersect with the area. While these techniques can be directly extended into 3D, they do not seem to work very well. For example, with the deficiency in depth perception, it is difficult to determine where to start the far corner of a selection box in order to enclose a certain object.

However, since the bat can be used for orientations in addition to positions, new techniques can be developed to facilitate object selection.

6.2.1 From laser gun to spotlight

We first experimented with a simple ray-firing selection mechanism, which we called "laser gun" selection. A ray is emitted from the bat along its X-axis, and the first

object it intersects becomes the candidate for selection. By rotating the bat with the fingers, the designer can easily aim at different objects and select them. There is virtually no need to move the wrist or the arm — an ergonomic advantage over mouse-base selection methods.

This method works well in most situations. However, when the target for selection is relatively small and far away from the center region, the designer may have difficulty in aiming due to the high angular accuracy required. This problem is often exacerbated by the fact that there is a certain level of noise in the orientation data [27]. To cope with this problem, we need a “softer” selector than a mathematical ray. This resulted in what we call the “spotlight selection” method. The idea is to decouple the requirement on angular precision from the distance between the cursor and the object. The spotlights used in theaters provide a good example. A spotlight does not have to point exactly on an character in order to shine on him or her, a certain level of inaccuracy can be accommodated with the spreading of light in a cone shaped area.

To simulate this effect, we first establish a selection cone with its apex at the bat and its axis being the X-axis of the bat (same as the ray of the laser gun selection). Objects within this cone are potential candidates for selection. Second, an anisotropic distance metric S is defined to emulate the spotlight effect: (a) if two objects are at equal Euclidean distance from the bat, the one which is closer to the center line of the cone should be selected, and (b) if two objects subtend a same angle at the bat with the center line, the one with the shorter Euclidean distance to the bat should be selected.

With the spotlight selection, the designer can select a distant object without having to aim at it perfectly, while selection of nearby objects is still as easy as with the laser gun method. It makes good use of both the position and orientation information of the bat, and avoids the disadvantage of measurement noise.

Since the graphics workstation we use supports spotlight lighting calculation, we use it as a visual feedback for selection (see figure 15). We also draw a transparent



Figure 15: A snapshot of spotlight selection.

cone to signify the selection area. The designer can see which objects are being lit by the spotlight, and can tell whether a certain object is close to the center line by the intensity of the light it receives from the spotlight. We use a white light for environmental lighting and a red light for the spotlight lighting, to show the difference. When the surfaces of the objects intersect with the cone, the intersection lines also provide information about their closeness of to the center of the cone.

6.2.2 Unification with 2D metaphor

After we used this spotlight selection method for a while, we noticed the disadvantage of using a fixed direction (with respect to the sensor) in firing the ray (i.e. along the X-axis of the sensor). The user often holds the sensor with the thumb, index finger, and middle finger, this makes the X-axis of the sensor more towards parallel with the screen than pointing into the screen. So when the user wants to select an object behind the cursor, he has to first rotate the sensor so it points towards it. In order to eliminate this type of rotation, we modify the spotlight selection such that whenever the user presses the button or key for object selection, the initial direction of the ray is always pointing into the scene. This unifies the traditional “under-the-cursor” selection technique with our 3D selection technique. When the user wants to select an object already close to the cursor, he can move the cursor in front of it and rely on the default direction of the ray to pick it. If the object is away from the cursor, then

the sensor can be rotated (with just finger movement) to capture it in the selection cone. No large wrist or forearm movement is necessary in either situation.

6.2.3 Implementation notes

There are an infinite number of distance metrics that satisfy the above mentioned two requirements for the anisotropic distance. In our current implementation, we use a metric that produces ellipsoidal isodistance surfaces (see figure 16). Suppose the spread angle of the selection cone is α , and $K = \sin\alpha$, the anisotropic distance between point $P = (x, y, z)$ and the origin is given by

$$|P| \frac{K}{\sqrt{1 + (K^2 - 1) * (x/|P|)^2}},$$

where $|P| = \sqrt{x^2 + y^2 + z^2}$ is the normal Euclidean distance.

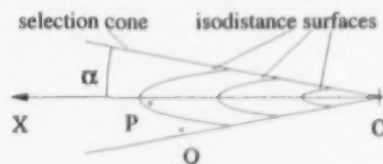


Figure 16: A cross section of the selection cone showing isodistance surfaces of the anisotropic metric S . Point P is considered closer to the origin than point Q .

The spread angle of the spotlight should match that of the selection cone (that is, α), to ensure a consistent visual effect.

6.2.4 Other feedback options

Our choice of a soft selector accommodates the special requirements of 3D object selection. However, it also introduces a certain level of visual ambiguity, as the closest object intersecting the cone may not be the candidate for selection. While this is intended and desirable in many cases, it can also cause surprise. To cope with this problem, we further refined the spotlight selection technique by adding more visual feedback. When the spotlight is on and the cone intersects with some object,

the current selection candidate is highlighted with a different color. This way, the user knows what object will be selected if he finalizes the selection. If the highlighted object is not intended, he can move or rotate the bat until the intended target is highlighted. This additional dynamic feedback further reduces the requirement for angular and positional accuracy and speeds up the selection task, as the user is relieved of the burden of judging which object is closest to the selector.

6.3 Graphic dials

Mappings from position change to orientation can sometimes be difficult to understand and use. In order to facilitate the input of intrinsically angular data, we designed the bat-operated graphic dial (see figure 17). For the same reason behind using the ring menu, the dial is coupled only with rotation of the bat around its X-axis, regardless of the overall orientation of the bat. In this way, the designer can start the dial with the bat in an arbitrary orientation and adjust it by rotating the bat around its X-axis, which is easy to execute.

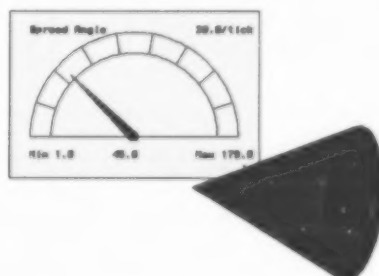


Figure 17: A bat-operated dial is being used to adjust the spread angle of a cone.

The rotation angle around the bat's X-axis can be calculated as follows. Suppose the bat's initial orientation is defined by three orthogonal unit vectors (X_0, Y_0, Z_0) , and its present orientation by (X_1, Y_1, Z_1) . To find out the rotation around the bat's X-axis, we first rotate (X_1, Y_1, Z_1) to (X_2, Y_2, Z_2) so that X_2 coincides with X_0 .

Then the angle we need is just the angle between Y_2 and Y_0 (the same as the one between Z_2 and Z_0). This rotation is uniquely determined by X_0 and X_1 : its rotation axis V being a vector perpendicular to the plane determined by X_0 and X_1 , and the angle θ being the angle between X_0 and X_1 . The cross product of X_0 and X_1 gives us V . Suppose V is normalized (a unit vector), we know that the rotation matrix defined by the tuple $\langle V, \theta \rangle$ is given by:

$$R = VV^T + (I - VV^T)\cos(\theta) + S\sin(\theta),$$

where I is the identity matrix, and S is the antisymmetric product:

$$\begin{bmatrix} 0 & -V_z & V_y \\ V_z & 0 & -V_x \\ -V_y & V_x & 0 \end{bmatrix}$$

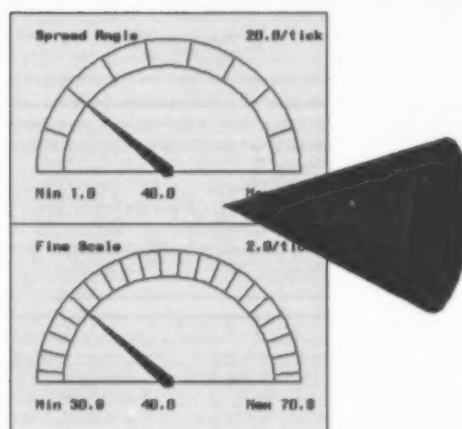


Figure 18: A secondary dial is used for higher precision.

In order to achieve higher resolution, secondary and tertiary dials may be used (see figure 18). Each new dial takes the current reading of the previous one as its center value and amplifies the angular resolution by a factor of 10. When used in conjunction with angular snap gridding, a relative precision of 10^{-4} can be readily achieved¹. It usually suffices for engineering use.

¹The usable angular resolution of a single dial is about 5 degrees.

For easy reading, the dials are always placed perpendicular to the viewing direction, and only translate with the 3D cursor. The plate of the dials are drawn translucently in order not to obscure the scene.

6.4 Chapter summary

Menu selection and object selection are two basic tasks in an interactive system. We discuss the special situation of performing these tasks in 3D with the help of 3D input devices, and show how iterative design process has been used to incorporate new experiences into our design. The evolution paths behind these techniques suggests that there is still plenty of room for further improvement. We also discuss the graphic dial as an angular input tool.

Chapter 7

Modeling Operations

For the quick prototyping of a solid model, a system must provide a set of operations for creating and manipulating solids. We pay special attention to providing smooth transitions among different modes, making good use of the 6D nature of the bat, compensating for its disadvantages, and offering easy access for frequently used operations.

The most important advantage of a bat over a mouse is the ability to create, place and modify objects by 3D direct manipulations. The fact that the designer is in the same space as the geometric model widely broadens the communication channel between the user and the system and relieves the designer from having to break down 3D tasks into 2D or 1D operations. This makes the interface “transparent” to a certain extent.

However, 3D direct manipulation is not just a trivial extension of existing 2D techniques. The added dimension and the nature of the input device, coupled with the display techniques, raise new challenges as well as offer new opportunities.

We discuss the challenges and opportunities, and new interaction techniques, in the context of three modeling operations: creation, placement, and reshaping.

7.1 Creation

A natural use of 3D input is the creation of 3D shapes by rubber-banding, as shown in [9]. In JDCAD, one creates a primitive solid by 3D rubber-banding its bounding box. The designer first chooses a shape from a ring menu, then presses a button to set the current position of the 3D cursor as one corner of the bounding box, the other corner of the bounding box follows the 3D cursor as it moves with the bat. When the solid is stretched to the right shape, the designer releases the button to finalize the solid.

While 3D rubber-banding is an extension to the commonly used 2D rubber-banding technique, there are issues unique with the added dimensionality. For shapes with three axial measurements (such as a box), the mapping from the bounding box to object geometry is straightforward. However, certain shapes have non-axial measurements (such as a circular shape) or more than three measurements (such as an extruded trapezoid). In those situations, proper mappings have to be devised.

Initially, we tried to employ the data of all six degrees of freedom simultaneously. It proved to be the wrong choice. We observed three facts. First, controlling all six degrees of freedom simultaneously tends to overload the hand. Second, orientation appears intrinsically more difficult to control than position. Third, mappings from an orientation change to a position change are usually difficult to understand and use.

Based on these observations, we adopt the following two rules.

First, if the shape in question has three measurements, but some of them are not axial in nature, we associate each one with an axial change. For example, a truncated cone has three measurements: the height, the diameter of the top, and the diameter of the base. We map the width of the bounding box to the top diameter, and the length of the bounding box to the base diameter. (The height of the bounding box obviously maps to the height of the truncated cone.) In this way, we make good use of the designer's ability to control 3D positions.

Second, if the shape in question has more than three measurements, we only directly control three of them, and compute the rest with a pre-defined proportion.

In this way, the designer can quickly get the object in roughly the desired shape and then come back to fine tune it. For example, an extruded trapezoid has four measurements: the width, the height, the bottom length, and the top length. We map the length of the bounding box to the bottom length, and keep the top length at a fixed proportion (e.g. the golden proportion) to it. The linear mapping makes it equally easy to set either the top length or the bottom length to a desired amount.

7.2 Placement

A large portion of the user interface in interactive solid modeling systems is devoted to the problem of placing and orienting objects in 3D with 2D input devices [5, 30]. Now with a 3D input device in hand, many complicated techniques can be replaced with simple and intuitive ones, at least in the sketching stage. Operations like grabbing an object and moving or rotating it with the bat provide kinaesthetic correspondence that is not available with 2D-based manipulations. However, such apparent gains in ease of use are not without cost.

We list the major challenges and ways to overcome them.

7.2.1 Position control

For sketching purposes, absolute numerical accuracy is not required in positioning an object, but relative accuracy, or precise relationship with other objects, is necessary. For example, different sections of an axle should be concentric, parallel sides of a box should align properly, and so on. Free hand manipulation alone cannot provide such accuracy. There are two ways to achieve such precise relationships, or *alignments*, (a) by snap dragging, and (b) by textual commands.

Multiple snap-dragging

Snap dragging uses a *gravity function* in direct manipulation to help the designer *snap* object points onto prominent features on existing objects (such as vertices,

edges, surfaces, and their intersections) [5, 16, 30]. For instance, one might translate an object through the displacement vector from a vertex on the object to a vertex on another object, thus aligning them at the vertex.

While such snapping techniques are very useful, they are not easily extendible into our interaction style. This is mainly due to two factors: the difficulty locating low dimensional entities, especially points, and the difficulty of proper visual feedback.

These difficulties in locating points can be attributed to input mechanism as well as output mechanism.

On the input side, there are two factors: tracker noise and hand vibration or unsteadiness. Tracker noise comes from various sources such as physical imperfection of the device, environmental interference, and limited precision calculations. Such noise in the tracking data causes the 3D cursor to jitter even when the sensor is perfectly still. Vibration or unsteadiness is a natural result of an unsupported hand. The combination of these two factors makes it relatively difficult to put and keep the cursor at a precise 3D location.

On the output side, there are also two factors: depth perception and occlusion. Even with all the depth cues supplied, depth ambiguity still exists. The use of occlusion for a depth cue, while generally effective, also makes certain parts of an object invisible to the user. These factors contribute to the difficulty in accessing low dimensional entities, especially points.

The difficulty in visual feedback mainly comes from the occlusion due to solid rendering, as the source or target points may not always be visible to the user. For instance, the center point of a box is always invisible to the user, no matter what the viewing direction is. So it is often difficult to determine whether a desirable snapping has occurred or not.

To cope with these difficulties, we modify the snapping technique to use multiple features and to offer effective visual feedback.

When the user moves an object with the bat, the system dynamically checks all possible snappings between the moving object and the set of reference objects. The

criteria for snapping are (a) all the possible snappings are accessible, (b) fluidity of motion is maintained.

To satisfy the first criterion, the use of gravity should not cause any pair of points from being excluded (masked out). For the second criterion, there needs to be a free moving zone between any two successive snappings, so the object is not forced to jump from one snapping directly to another.

According to these criteria, we design the snapping technique as follows. First, we designate a set of features points to be used for snapping. These include vertices, midpoints of edges, centers and quadrants of circles. Suppose the moving object M has a set of feature points $\{P_1, P_2, \dots, P_{n_p}\}$, where $P_i = (P_i(x), P_i(y), P_i(z))$, and the combined set of feature points from reference objects are $\{Q_1, Q_2, \dots, Q_{n_q}\}$, where $Q_j = (Q_j(x), Q_j(y), Q_j(z))$. Snapping from P_i to Q_j occurs along an axis, say the X -axis, when the following criteria are satisfied:

1. $\{P_i, Q_j\}$ is the minimum distance pair along the axis,
2. the distance between the pair D_s , in this case $D_s = P_i(x) - Q_j(x)$, is less than a threshold τ , that is $D_s < \tau$, and
3. the distance D_s is smaller than half of the next smallest distance between any other pairs.

The last condition guarantees a free moving zone between any adjacent snappings.

This snapping technique frees the designer from the burden of picking specific points for alignment. The designer can perform alignment without lowering the level of abstraction from 3D objects down to points.

To offer an effective visual feedback, we draw a plane-like object at the snapping position, perpendicular to the axis along which the snapping occurs. The appearance of this plane serves both as a confirmation of snapping and as an indication of the actual location of the snapping.

In order to maintain visual continuity and to provide a measurement of error, we use a wireframe stand-in for the object being moved. It always follows the 3D cursor

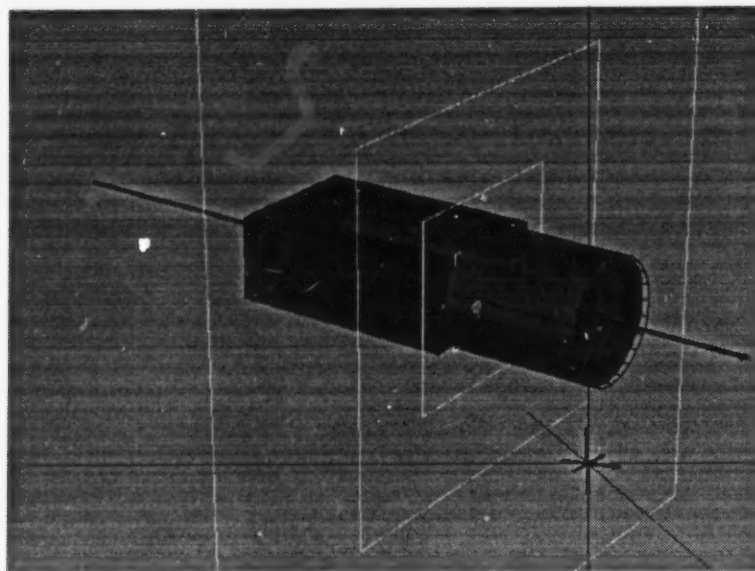


Figure 19: Active snap grid.

with free hand motion. The solid rendering of the object is shown snapped according to the closest alignment.

In figure 19, the user is translating the cylinder along the active axis, with the box being the reference object. The light wireframe plane indicates that the center of the cylinder is snapped to the front end of the box. The wireframe stand-in of the cylinder indicates the natural position and orientation of the cylinder without snapping.

Since objects maintain their shapes during placement, implementation of the multiple snap dragging technique can take advantage of this fact. One can collect all the feature points on an object and store them with the object as sorted lists. At the beginning of snap dragging operation, the feature points from all the reference objects are transformed into a single coordinate system (the reference frame), and sorted into one list per axis. We call these lists the source lists. The feature points from the moving object are also transformed into the reference frame and sorted into lists accordingly. We call them target lists. This is done only once — at the beginning of the operation — after that, their new locations can be found by adding the current

displacement vector, that is, the displacement vector from the starting position of the bat to the current one. The time it takes to prepare these two lists is linearly proportional to the total number of points in each lists, as each step requires a merging of two pre-sorted lists. The task for finding the current minimum distance pair along an axis can be accomplished by a merging of the two lists according to the current displacement vector. This is also a linear complexity task, as both lists are sorted. The linear complexity nature makes this technique scalable.

Command-based alignment

Command-based alignment operations serve as complements to snap dragging techniques. While they are less intuitive than direct manipulation, they can save time for experienced users. For this purpose, we provide a set of commands for various alignments.

The new challenge here is the continuity in visual feedback. If an alignment command moves the object over a long distance, the user may have difficulty understanding the effect, as the object would appear to jump from one place to another. To provide a smoother transition, we employ animation techniques to visualize the process. When the user issues an alignment command, the object moves from its original place to the target position in a number of steps, n , (usually $n = 20$) over the duration of one second, which gives the illusion of a smooth motion. This smooth motion helps to visualize the effect of the command, and relieves the user from having to mentally re-assimilate the scene after an object jumps across.

Discussion

One possible trade-off between accessibility of points and solid occlusion is transparent (translucent) rendering. However, this technique is computationally expensive when sorting and splitting is required (with the current graphics hardware). Also, it does not scale well, as its effectiveness quickly degrades when the number of transparent layers increases. The usable number of layers is only about 3 or 4, even when different

colors are used. A more powerful rendering method is needed for providing visual access to all the points while supplying an effective depth cue.

7.2.2 Orientation control

Orientation control in 3D is often more challenging than position control. 3D rotations are non-commutative. So it is difficult to compose a rotation from several rotations, say using Euler angles. This is the main reason why 2D pointing devices and 1D dials are not very effective in controlling 3D orientation. With the 3D orientation input from the bat, however, the user can now execute a rotation intuitively, by specifying rotation axis and rotation angle simultaneously. It also allows for continuous change of the rotation axis. This is the main advantage of having a 3D orientation input.

Just as in the case of position control, free hand control alone is inadequate. The most useful rotations in engineering design are those constrained in some ways. However, the orientation space does not have an intrinsic uniform grid. This makes it difficult to provide snapping for orientation except for a few simple situations.

Axis selection

Many placement operations require the selection of an axis. For example, the designer may want to slide an object along one axis, without jeopardizing placement along the other two axes. This type of constraints can be specified by keyboard commands or menu selection. However, that requires the user to know the name of the axes at the time of the operation. With the orientation data from the bat, we can make this task much easier and more intuitive. The user can point the bat roughly at the direction of the intended axis and instruct the system to find it. This way, the user does not have to remember which direction is associated with which axis.

This is probably the simplest snapping in orientation, yet it is very effective in eliminating named axes for the user.

Octant snapping

In the scope of mechanical design, the directions along the three major axes are often preferred. For this reason, we design an orientation snapping according to all 24 octant-aligned orientations, i.e. ones that transform each of the three axes to be parallel or anti-parallel to an original axis. These rotations are provided as direct manipulations. The user rotates the object with the bat, and the system selects the closest match between the freehand rotation and the pre-computed set of octant-aligned rotations. The user does not have to identify a rotation axis in advance, thus making the operation very intuitive.

We use distance in quaternion space as a metric in comparing orientation differences. The pre-computed orientations are stored as 24 unit quaternions, and the incoming orientation is also expressed as a unit quaternion. We take the dot products of the incoming quaternion with each of the 24 quaternions, and the one producing the largest value is taken as the result.

Rotation angle transfer

The second type of constrained rotations are ones around a fixed axis. This provides rotation control through Euler angles, if we use the three major axes. The user first selects an axis by the bat, then rotates the bat to control the rotation angle. Since the rotation axis of the bat may not be the same as the selected axis, there needs to be a transfer of rotation angles. For intuitiveness, this transfer must satisfy several conditions.

1. If the rotation axis specified by the bat coincides with the rotation axis of the object, then the full amount of rotation angle from the bat should be transferred to the object. Likewise, when these two axes are anti-parallel, the same angle with a different sign should be transferred.
2. When these two axes are perpendicular to each other, no rotation should be transferred to the object.

3. The transfer function should be monotonic and smooth between the cases stated above.

A cosine function is well suited to these conditions. If the angle between the two rotation axes is θ , and the rotation angle from the bat is α , then the rotation angle transferred to the object is $\beta = \alpha \cos(\theta)$.

This transfer function ensures a smooth and intuitive control of the constrained rotation with an unconstrained input. A grid on the rotation angle can be enforced to provide accuracy on a coarse level (i.e. snapping to a multiple of 5 degrees). To specify accuracy to the degree level, the graphical dial technique discussed in section 6.3 can be used. This is usually sufficient for prototyping purposes.

7.2.3 Section summary

In this section, we discuss a very important task in 3D modeling — the placement of an object in 3D space. For position control, we are confronted with the difficulty of accessing certain points on objects, mainly due to the input and output methods. A multiple snap dragging technique is designed to cope with the problem, together with new feedback methods. We also discuss the use of animation techniques to provide better feedback for command driven alignment operations. For orientation control, we discuss the use of snapping on orientation input from the bat and the intuitive transfer of rotation angle between different axes.

7.3 Reshaping

Modifications to a model are at least as important as its creation, and usually occur much more frequently. We design two types of reshaping operations to facilitate this task. We first discuss a region-based direct manipulation operation to take advantage of the bat. We then discuss ways of supporting precision in reshaping operations.

7.3.1 Region-based reshaping

In an ordinary 2D drawing system, the reshaping operation is often based on a set of "handles" on the object. The user drags a handle with the mouse to enact a certain type of reshaping (e.g. stretching, vertex relocation, etc.). Although this method can be directly extended into 3D, we found it less effective than in the 2D case. First, adding a third dimension increases the number of handles required to accommodate the types of reshaping operations. So, each handle can not take up too much space, otherwise, they will obscure the object itself. However, the measurement noise and lack of sufficient depth cues makes it a difficult task to place the 3D cursor at an exact handle point, as we discussed in the previous section.

In order to resolve this dilemma, we use an alternative way of visual feedback in reshaping — a region-based reshaping metaphor. The idea is to extend the handles into 3D regions, and make them invisible. The selection of region is indicated by the shape of the cursor, instead of its proximity to a handle. Conceptually, the space around the object is divided into disjoint regions with each one corresponding to a different type of reshaping operation. After the designer selects an object, he or she moves the 3D cursor into one of the regions before commencing the reshaping action. As visual feedback, the 3D cursor takes different shapes in different regions. When the designer presses a key to start reshaping, the system attaches certain measurements of the object to the the 3D cursor, and the object changes shape as the bat moves. Releasing the button finalizes the shape.

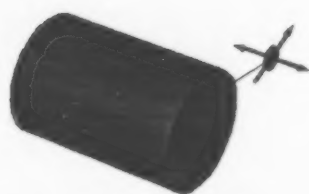
Composite objects can be reshaped via their bounding boxes, in one, two, or three dimensions. If the starting position of the cursor is near the center of a face, then the object will be stretched only along the dimension perpendicular to that face. If the cursor started near the mid point of an edge, then only the two dimensions perpendicular to that edge will change with the cursor. Similarly, the designer can pick a corner and change all three dimensions at the same time.



(a) Ready for height change



(b) Ready for radius change



(c) Ready for height and radius change



(d) During reshaping

Figure 20: Region-based reshaping.

7.3.2 Support of precision

Just as in the case of position and orientation control, freehand manipulation is useful but inadequate. We provide three mechanisms to support precise reshaping.

First, a snap grid can be used to control the creation and reshaping of an object. This ensures that the dimensions of an object are always multiples of the grid size.

Second, reshaping through conformation operations. A conformation operation changes the size and shape of its target object according to reference objects. For example, a hemisphere can be made to have the same radius as a cylinder, so it can form a smooth cap to one end of the cylinder.

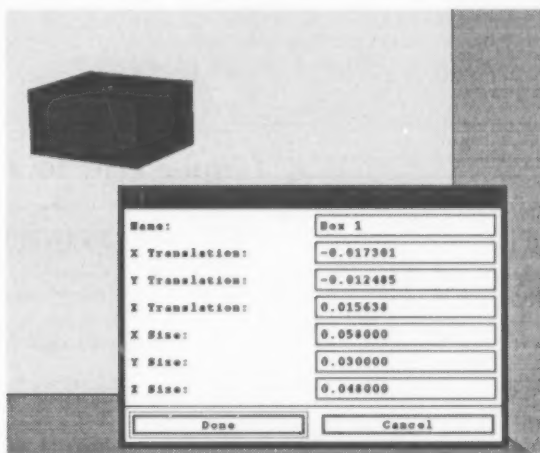


Figure 21: A dialogue box for shape editing.

Third, textual input. The user can invoke a standard X window dialogue box to set various parameters of an object, including its size. This provides the designer with the ultimate expressive power in terms of precision. Figure 21 shows the dialogue box for editing the position and size of a box. This is a good example of the advantages offered by a non-immersive style, as the user has free access to the mouse and keyboard, and all the interaction techniques associated with them.

7.4 Bat-based clipping

In designing a complex model, the designer often needs to inspect the internal structure of a certain part (e.g. an assembled component). Cross sectioning is the traditional method for this purpose. We designed a bat-based clipping tool to further facilitate such operations.

Since the SGI IRIS VGX workstation supports six additional clipping planes in rendering, we use them to form several clipping tools the designer can dynamically place in space with the bat. So the designer can "poke around" to examine internal structures and profiles of cavities.

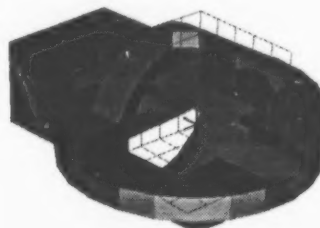


Figure 22: A clipping tool is being used to explore the internal structure of a casing.

In our current system, the designer can dynamically clip out a half space, a quarter space, or an octant (as in figure 22). When more than one clipping plane is used and they form a convex cutting volume, multiple passes of rendering are necessary, as each clipping plane cuts out a half space. These clipping tools are also useful for interrogating a model under simulation, for example, collision checking for a simulated robot arm motion.

Rossignac et al developed a solid-clipping algorithm [32] for similar purposes. Their algorithm is more sophisticated and more expensive in computation.

7.5 Chapter summary

In this chapter, we discussed modeling operations such as creation, placement, reshaping, and inspection of objects. While many of the interaction techniques are extensions of 2D ones, the characteristics of the input device, the display method, and the new interface style raise new challenges and offers new potential. We discuss several interaction techniques aimed at overcoming the difficulties and making use of the potential. While these techniques are developed in the context of 3D solid modeling, they also provide a reference in designing other types of 3D operations.

Chapter 8

Evaluating JDCAD

Since the main purpose of this research is to increase efficiency in 3D solid modeling, especially in prototyping 3D models, one way to verify the results is to demonstrate a reduction in modeling time by comparison with existing systems. However, there are many practical difficulties involved in making a fair and meaningful comparison. Nevertheless, we carried out an experiment to show the potential of the new interface style. Another way to verify the merits of the new style is by comparing it with the traditional pencil-on-paper sketching, as it is the norm of current practice and also the ultimate goal of the research — to make a computer capable of competing with “used envelopes”. We discuss observations made during these two experiments and their implications.

8.1 Comparing with a commercial CAD system

8.1.1 The challenges

With the nature and scope of this research comes the challenge of evaluation. There are several major reasons why a fair and accurate evaluation is difficult to achieve.

First, since JDCAD is a prototype of a new style, there is no similar system to compare with. As we have indicated in the survey part, there are only three related experiments in the past, namely, Clark's pioneering work at Utah, Sach's 3-Draw

at MIT, and Butterworth's 3DM at UNC. The first one, Clark's system, was meant for 3D curved surface editing. The second one, 3-Draw, was designed for 3D curve (polyline) editing. Neither one has provision for rapid prototyping of engineering designs. The third one, 3DM at UNC, is the closest in terms of modeling capability, as it allows for the creation and manipulation of 3D regular shapes. However, as the implementation of 3DM is strongly attached to the PixelPlane 4 system (a unique high performance graphics supercomputer designed and built at UNC), it is virtually impossible to conduct a comparison locally. In short, no similar systems are available for comparison.

This forced us to choose a commercial CAD package for comparison, which resulted in the second problem, that is, a major difference in the amount of development effort. Most commercial CAD packages have hundreds of staff-years of development effort behind them, if not more. To this end, a single person project like JDCAD could never come close. This qualitative difference puts our prototype system in a very disadvantageous position. A commercial CAD package usually has a large set of commands, including shortcuts for frequently used command sequences. It also provides a larger set of primitive shapes for easy access. These options are not available in a skeletal prototype like JDCAD.

Due to practical reasons (mainly the availability of CAD packages and test subjects), we chose Autodesk's AutoCAD system as the reference package. AutoCAD is a widely used CAD package on microcomputers. It was originally designed for drafting, but 3D capabilities were introduced in later versions, making it capable of supporting full 3D design. In our evaluations we use release 12, running on an Intel 80386-based PC, with SoftEngine 386 (a display enhancement software from Vibrant Graphics).

8.1.2 The first test object

The first task in designing this experiment is to choose a 3D shape as the test object. We used the following shape for our first experiment. It consists of five primitive

shapes and four boolean operations. As this is for a prototype, test subjects were told that absolute dimensions were not important as long as the relative proportions were roughly presented.

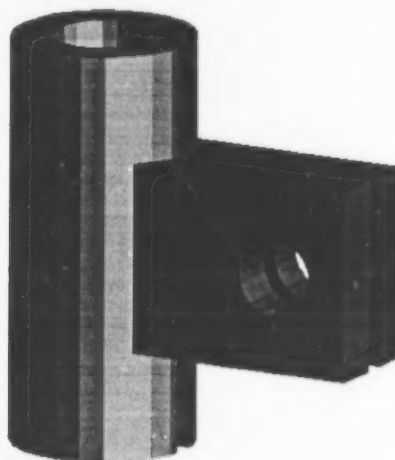


Figure 23: The first test object.

We pick this shape because it is easily achieved in both systems and uses just two common primitive shapes (box and cylinder). This shields us against the difference in primitive repertoire. The simplicity is also important for minimizing the effect of command set difference.

8.1.3 Using a 2D-based System

When modeling the test object, the first thing to do is to plan on several issues. First, since there is no intuitive way of changing viewing parameters, one usually has to set up several views, each at a different angle. A common setting is to have three orthogonal views (e.g. front, top, and right) and a pictorial view. This corresponds to the common practice in engineering drafting. There are several advantages for setting up multiple views. First, since most CAD systems allow a different mapping for each view, the designer can have access to different combinations of 2D directions

by changing the active view. Second, projections from different directions can provide more spatial information about the 3D object. However, there are also disadvantages to the multiple view setting. First, the real estate problem. Since screen space is a constant and has to be split among all the views, the effective area for each individual view is limited. Second, the correspondence problem. Since the designer has to mentally correlate all the views into a single design space, there is a cognitive load. This becomes more of a problem when each view is at a different scale, so the designer not only has to do mental translation and rotation, but also mental scaling. Due to the real estate problem, zooming in on a view is often needed. However, there is usually no dynamic link among the scaling (zooming) factors of the views. It often happens that each view is focused on a different part of the model, and getting the whole picture becomes difficult. When 3D direct manipulation of view parameters is available, a single view is usually adequate. This avoids the problem of screen space contention as well as scaling discrepancies.

The second step in designing a 3D object in a 2D-based system is to establish the necessary construction lines, which will facilitate the modeling process. While it is possible to do modeling without construction lines, they usually make the job easier and faster. This is due to the 2D nature of the interaction — that the cursor is grounded on a 2D plane. For example, in order to move a circle off the plane on which it was created, the designer has to either specify the 3D translation by typing coordinates (whether absolute or relative), or make its reference point coincide with an existing point (say, one end of a construction line). While creating construction lines acts as a time saving step for the whole design process, it is a distraction to the designer in the prototyping stage. Instead of focusing on the shape of the object, the designer has to consider how to place certain points in space, which can be used to attach shape elements later on. Usually this forces the designer to do a sketch on a piece of paper to identify the necessary construction lines, then move to the computer to carry out the drawing. For our test object, this step of planning usually takes a few minutes.

The actual drawing of construction lines also poses a problem. Within each view, the cursor moves on a 2D plane (the current XOY-plane). So the designer has to type in the coordinates of a point in order to get into the third dimension. This requires the designer to be aware of the naming of the axes in each view. For example, a point visually above the XOY-plane may have a positive or negative Z-value, depending on the way the current coordinate system is defined,

Another problem with this 2D-based interface style is the visual inconsistency regarding depth. If a user specifies a point by directly pointing with the cursor, the point is always on the current XOY-plane. However, if the designer chooses to snap the point to a certain feature (say, an end point of a construction line), then the point may well be off the current XOY-plane. While this is the intended way of using construction lines and such, it is visually confusing, as the designer has to remember which features are on the current plane and which aren't. Without good depth visualization, this is often a quite difficult and error-prone task.

When it comes to the actual drawing of the object, the process is slowed down due to the switch among different user coordinate systems. Since the cursor moves on the XOY-plane of the current coordinate system, direct manipulations are constrained to that plane. To do direct manipulation in the third dimension requires switching to a different coordinate system. For example, the designer may create a rectangle on the current plane by 2D rubber-banding. However, to drag it in the third dimension requires a coordinate system whose XOY-plane is perpendicular to the current one. Since there is no visually intuitive way of specifying a user coordinate system, the designer has to name each coordinate system carefully to be able to retrieve them by name.

Another common problem associated with a 2D-based system is the differentiation between the XOY-plane and the third dimension. For example, in order to create a 3D box, one often has to draw a 2D rectangle first, and then assigns a "thickness" to make it grow into a 3D shape. While creating a rectangle is often interactive (e.g. by using rubber-banding), assigning the thickness is often text-based and thus lacks the

kineasthetic correspondence. In prototyping, this means the designer no longer has direct manipulation over the proportion of all three sides at the same time.

Similarly, while the designer may select the box entity and interactively stretches it on the XOY-plane, he usually cannot do the same to the thickness. This type of differentiation among the three dimensions creates extra difficulty for the designer, as he has to invoke different operations for visually identical tasks.

During the experiment, we also noticed the effect of a 2D-based metaphor on the level of abstraction of a model. Since most 3D systems are developed from their 2D-based predecessors, they usually inherit their notions from the line-based drafting process. For example, the designer can snap a point to line-based features such as an end point, intersection point, or mid-point. However, there is no provision for a "mid point" of an object along an axis. In order to align two rectangles along the Y-axis, the designer may have to draw a horizontal line through the center of one rectangle to provide as a reference line. This inconvenience is not the direct result of 2D-based metaphor, but a reminiscent of the drafting-oriented design concept.

8.1.4 Using JDCAD

Using JDCAD, the test object can be created in just a few steps. First, the designer pops up the ring menu to pick the cylinder primitive.

He then creates a cylinder by 3D rubber-banding. Since both the radius and the height are under direct manipulation, it only takes a few seconds to establish the approximate dimensions. Note that the designer does not have to worry about two separate aspects of the object (one being the radius on the XOY plane of the current User Coordinate System, or UCS, and the other being the elevation).

The cylinder can be readily positioned to the desired position in the space, and the view adjusted to a suitable angle. In addition to establishing a proper view, the direct viewing adjustment also serves to verify the resulting shape of the object. This becomes more important when the object takes a more complex shape. Compared with a system based on multiple 2D projections, this is one of the most important ad-

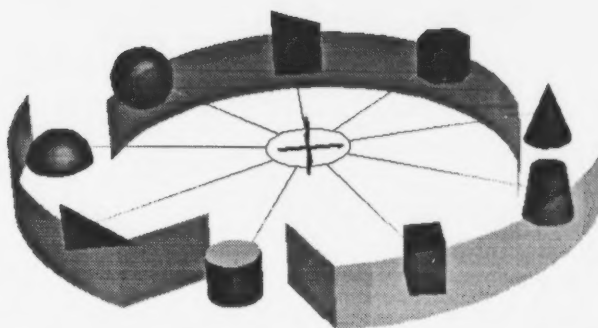


Figure 24: Choosing the cylinder primitive with the ring menu.

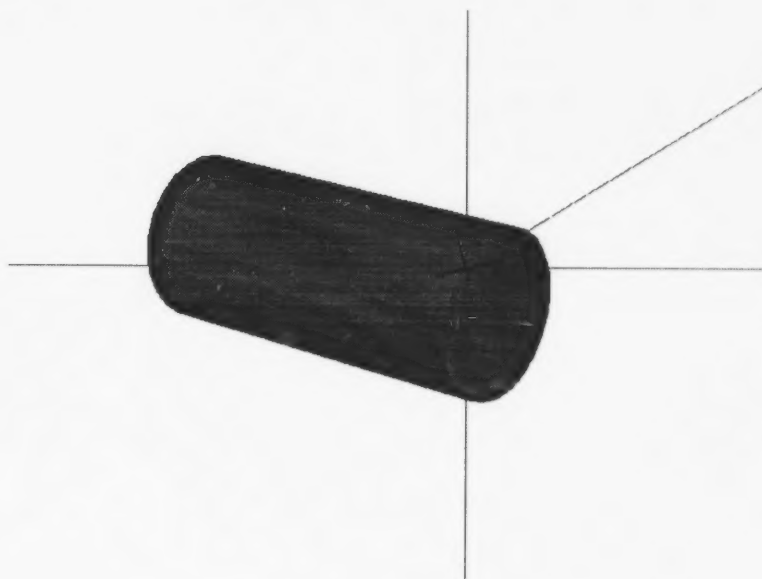


Figure 25: First Cylinder created.

vantages, as the shape of the objects are revealed instantaneously when they undergo a directly controlled motion.

The designer then creates another cylinder to drill a hole in the previous cylinder. Since the objects are all in 3D, doing a "centering" operation in all three directions is a very intuitive alignment operation. Since most 2D-based systems focus on lower level entities (such as line segments, circles, etc.), they usually do not provide these types of 3D-based operations. This is one example in which the elevating of the abstraction level reduces modeling time.

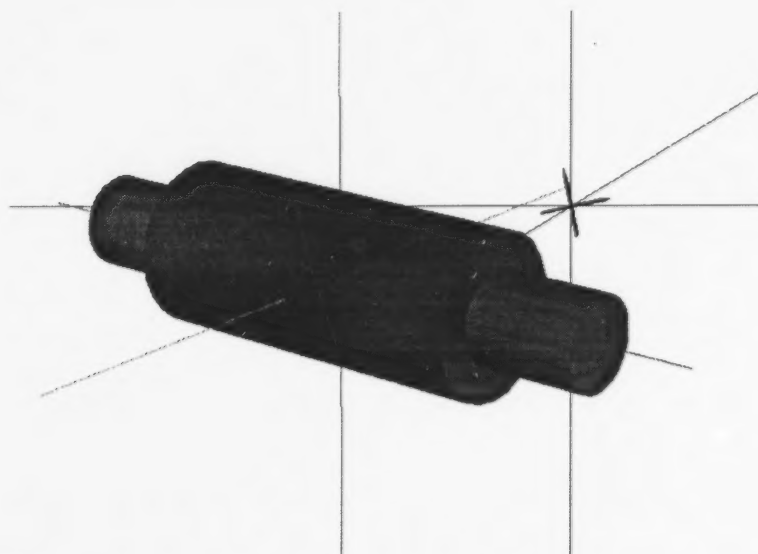


Figure 26: Second cylinder created and aligned to the first one.

After performing a subtraction operation, a "pipe" shape is formed. Since this shape is very common in engineering design, one may just provide a "pipe" primitive to the designer instead. In such a case, the user will just rubber-band the outer radius and height simultaneously, with the inner radius linked to the outer radius, say, by the golden proportion. When the outer shape is settled, the designer comes back to adjust the inner radius, if necessary. This is the case in JDCAD version 1 and JDCAD+.

Since the gap is uniform from inside to outside, one can create the ears as one

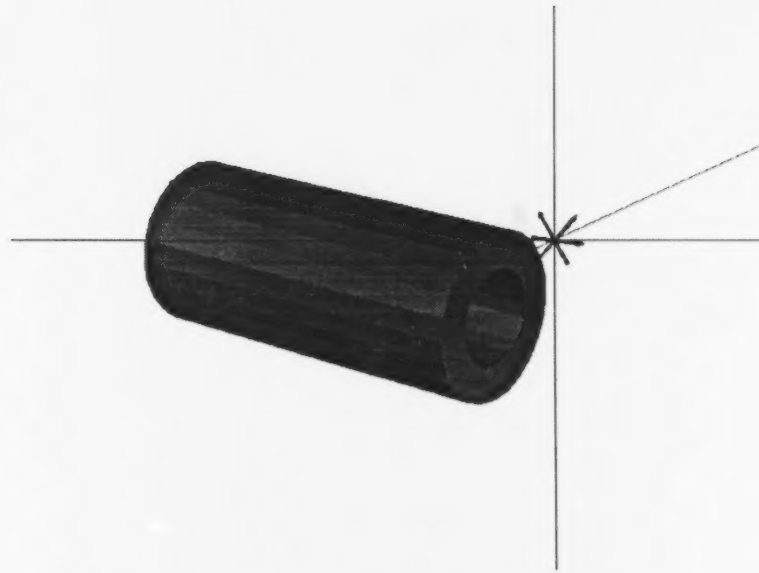


Figure 27: A pipe formed by "subtract" operation.

solid, and then split both the pipe and the ears later. With this in mind, the next step in modeling is to create a box for the ears. This is easily done by picking the box primitive from the ring menu and rubber-banding it to approximately the right size. Alignment with the pipe is easy, as all three directions are accessible at the same time, and the designer can pick an axis by simply pointing the bat at roughly the same direction. Here, 3D direct manipulation offers two handy aids. First, we achieve a certain level of "name-transparency" for picking an axis. The designer does not have to worry about which axis is which, he only needs to give a rough approximation intuitively. Second, the relaxation in orientation accuracy reduces the aiming time required. In its 2D-based counterpart, a similar reduction can be achieved in 2D (say, by turning on "ortho" mode), but not in all three directions.

A union operation welds the box to the pipe, making it ready for the holes on the ears and the gap. The designer creates a second box, a thin, wide, and long plate, as the cutting tool for the gap. As the plate should be quite thin, it is often easier to activate the shape editing dialogue box (cf. figure 21) than to use rubber-banding. This shows the advantage of a non-immersive interface over an immersive one, in the

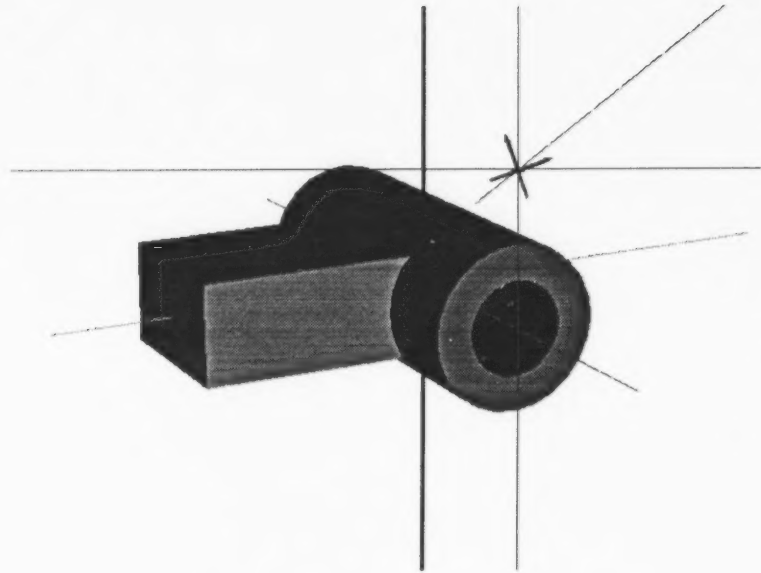


Figure 28: A box formed and centered to the pipe with respect to the selected axis.

case of the latter, one is usually kept from using the keyboard.

After the gap is formed, the designer then proceeds to drill a hole through the ears. This can be done by first creating a cylinder as he did before, then place it at the desired location. Since all six degrees of freedom are available for manipulation, this can be done in a single motion.

The active snap grid relieves the designer from having to be exact. He then subtracts the cylinder from the object and forms the holes. This completes the modeling of the test object.

Between each step of the process, the designer can easily check the shape from any angle simply by rotating it in his hand, and poking inside using the clipping tools. As such, he never encounters any ambiguity about the 3D shape.

Since the objects are all in 3D, they are sufficient in themselves to specify a reference coordinate. Furthermore, the naming of the axes is transparent for the designer, relieving him from having to remember which axis is which and which direction is positive or negative. The level of abstraction permits the use of features for active alignment, thus reducing the need for extensive preparations of auxiliary

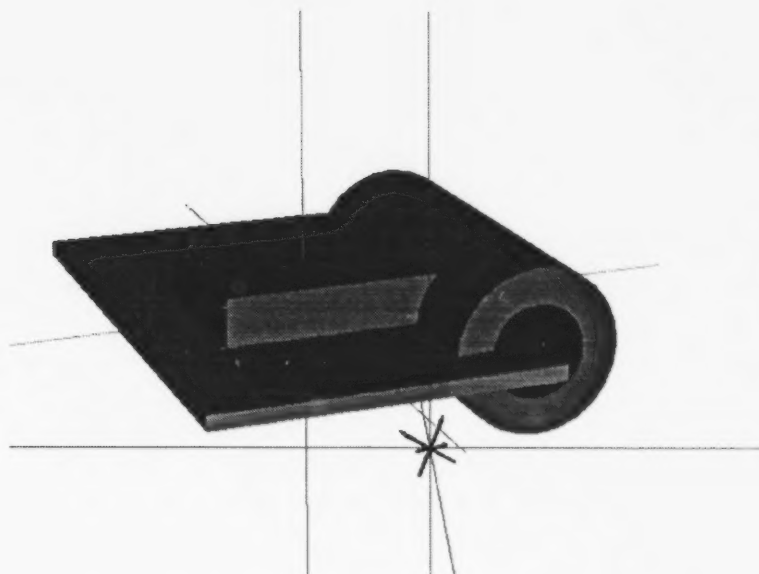


Figure 29: A plate created and aligned to the object.

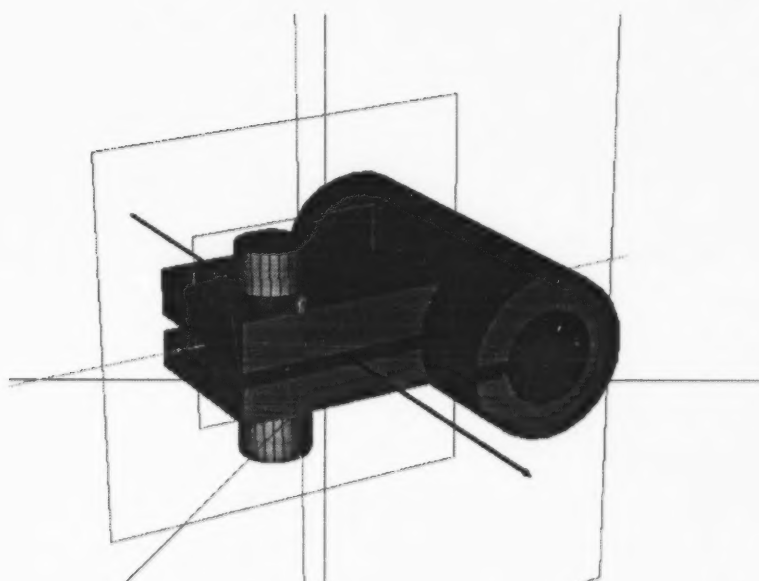


Figure 30: A third cylinder is being positioned with active snap grid.

structures. These factors contributed to the overall reduction in modeling time.

8.1.5 The second test object

The first test object is quite simple. In order to test the systems on a more complex object, we designed the second test object. It consists of 14 primitive solids and requires 6 union operations and 7 subtraction operations.

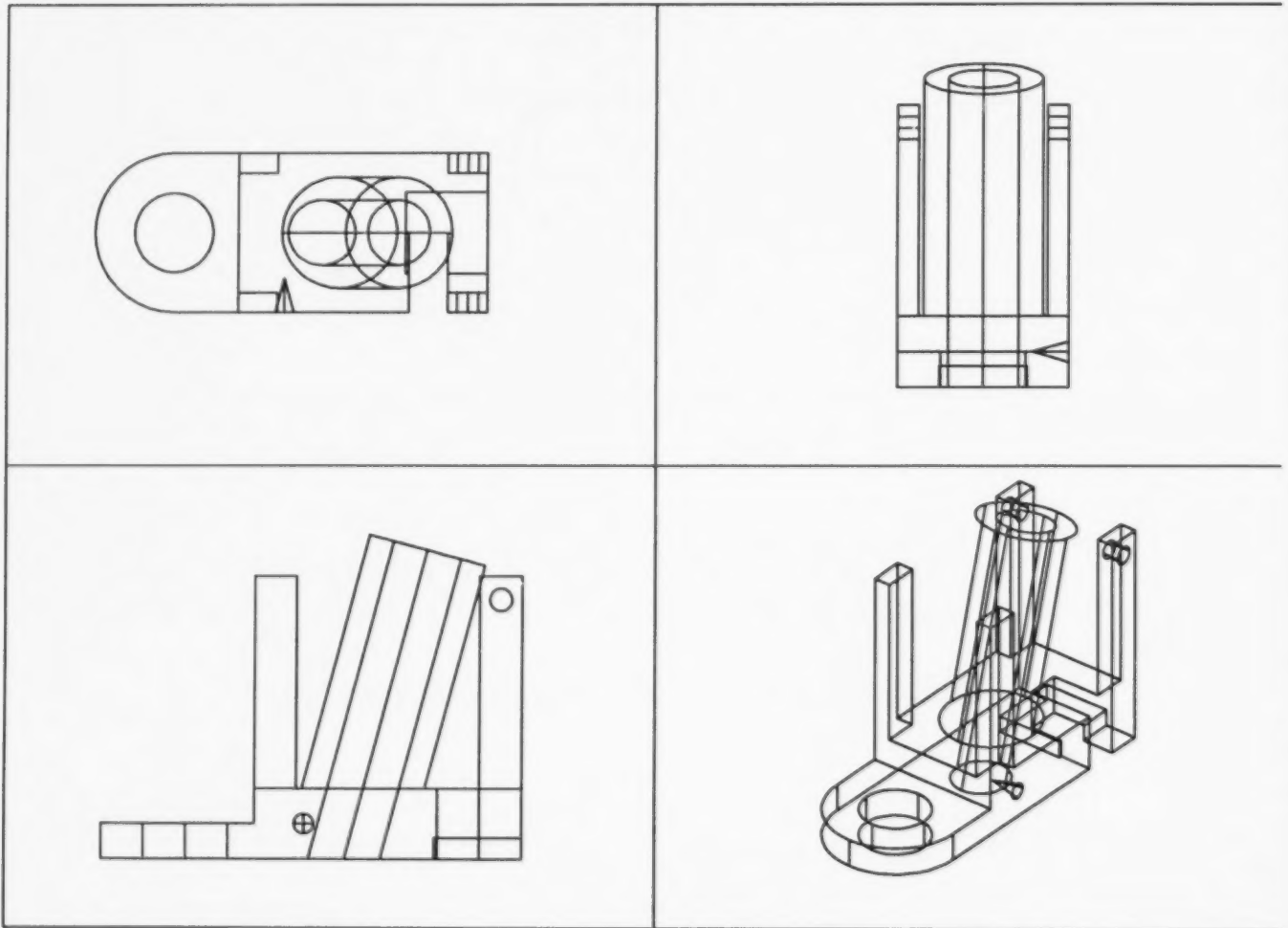


Figure 31: The second test object in AutoCAD wireframe mode.

8.1.6 Discussion of the timing results

Due to limited resources available to the author, the tests were performed only in a small scale. Three subjects, two males and one female, took part in the tests. All three subjects were students from an AutoCAD class but with different backgrounds. Each subject was given about 8 hours to become familiar with JDCAD, and then built each test object three times, both in AutoCAD for DOS and in JDCAD. AutoCAD version 12 with the Advanced Modeling Extension (AME) release 2 was used for the test. The PC platform was a 16 MHz 386SX with a 80387 math coprocessor and 4 Megabytes of RAM. An SGI IRIS/310 VGX machine was used for JDCAD.

The following tables contain the time measurements. The measurements were rounded to the nearest 5-second interval. Figure 32 shows the comparison.

Table 8.1: Timing for the first test object.

	Using AutoCAD			Using JDCAD		
	1st	2nd	3rd	1st	2nd	3rd
Subject A	12:55	11:20	10:35	5:30	4:35	3:05
Subject B	17:25	15:30	14:05	3:20	2:50	2:15
Subject C	15:35	13:10	12:15	4:55	3:30	2:45

Table 8.2: Timing for the second test object.

	Using AutoCAD			Using JDCAD		
	1st	2nd	3rd	1st	2nd	3rd
Subject A	38:05	35:15	33:05	10:30	9:05	7:20
Subject B	50:40	45:05	43:15	9:30	7:15	6:35
Subject C	46:05	41:15	40:45	13:30	8:55	6:20

The difference of drawing times is apparent from the graphs. It shows that JDCAD is much faster than AutoCAD in designing the two test objects.

Although this is a very encouraging result, there are reasons one should be cautioned about its nature.

First, there is a significant difference between the two hardware platforms. This is mainly due to the lack of resources. The difference in computational power favors

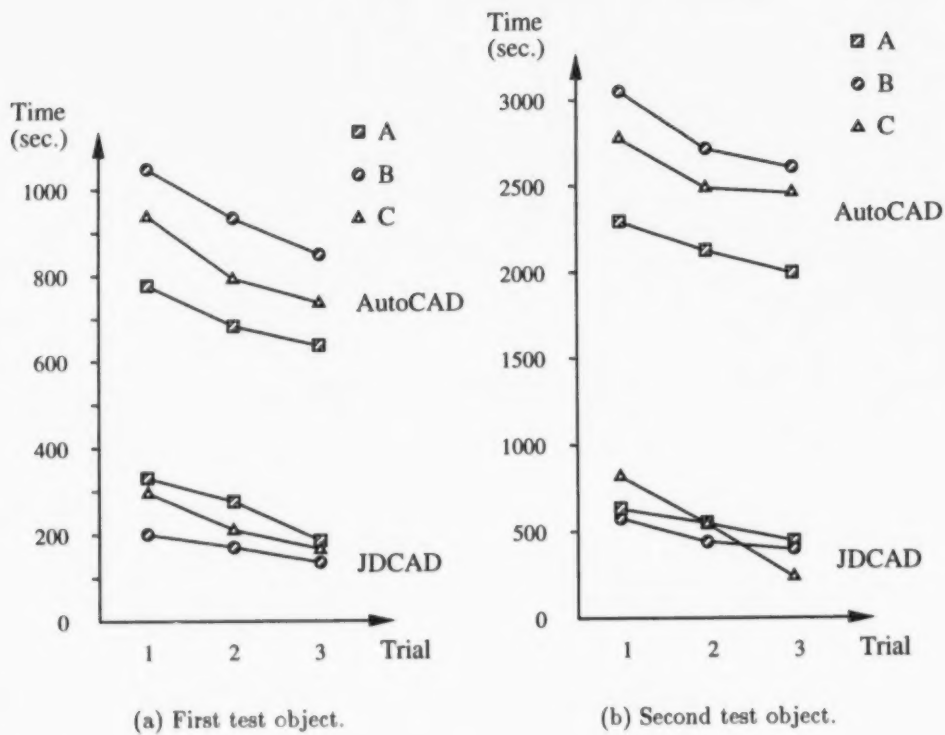


Figure 32: Comparison of drawing times.

JDCAD.

Second, the two CAD environments are very different in their purpose, amount of development effort, complexity, and functionality. Such differences make the comparison less than fair. This difference in development effort favors AutoCAD.

Third, there is no known effort in setting up a set of benchmark objects for the comparison of drawing speed for CAD systems. Each discipline of engineering has its own set of frequently used shapes. A CAD system tailored for a specific discipline or area would certainly perform better, in its intended discipline or area, than a more general purpose system.

As for subjective preference, all test subjects preferred the interface style of JDCAD, as it offers a much better visualization of 3D objects during the design, and allows for a much more intuitive way of manipulating them.

8.1.7 Section summary

In this section we discussed an experiment comparing the prototype system JDCAD with a typical commercial CAD system (AutoCAD). Two sample mechanical designs were used to compare the advantages and disadvantages in using both systems for prototyping purposes. The emphasis is on the ease of use during the design process. It is shown that the new style of 3D direct manipulation embodied by the JDCAD system possesses a significant advantage in this respect. This advantage is achieved by promoting the level of abstraction from 1D and 2D entities to 3D entities, by the intuitive interaction techniques, and by the inherent advantage of 3D direct manipulation.

Timing data is also collected. It shows that prototyping with JDCAD takes much less time than that with AutoCAD. There are several factors that limit the accuracy and fairness of the comparison, and the results should be taken as anecdotal evidence only. Nevertheless, it gives a rough measure of comparison, and indicates the potential of the new interaction style.

8.2 Comparing with hand sketching

As we discussed in the previous section, there are a number of challenging problems involved in making a fair and meaningful comparison between the new interface style and the traditional one. However, there is another way to test the effectiveness of the new interface style — by comparing it with hand sketching. After all, our goal of research is to extend the use of CAD technology into the very early stage of the design process — sketching, which, up to now, largely remains a manual process. If we can show that a designer can “sketch” out a model in our style in less time than with manual pencil-and-paper approach, then the advantage of our style is evident.

8.2.1 Designing the experiment

We asked students from a design class in the Industrial Design Studio of our university to participate as volunteer test subjects. Ten test subjects, five males and five females, participated in the test. Each test subject was first introduced to the purpose of the test, and explained the type of sketch he or she would make. The drawings were required to be at the level of concept sketch, containing enough detail to communicate a general idea about the 3D shape in question, but not too much details to fall behind the train of thought during conceptualization. Each test subject was asked to make a sketch for each test shape with his or her normal drawing speed. Before making the sketch, the test shape was called up into JDCAD and inspected by the test subject. After the shape was well understood, the test subject would start a timed sketching of the shape. Once the sketch was done, it was assessed for sufficiency in detail and examined for possible discrepancy with the test shape.

8.2.2 The test objects

Three mechanical shapes created in JDCAD were used as test shapes (see figure 33, figure 34, and figure 35).

We include a few samples of the hand sketches made by the test subjects (see

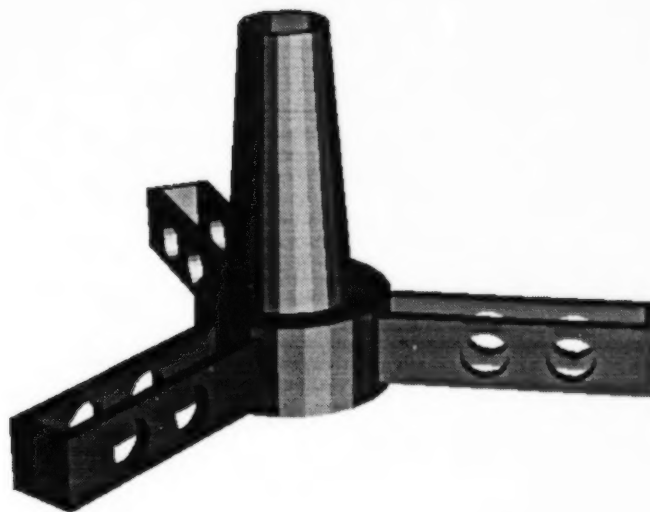


Figure 33: Test object A.

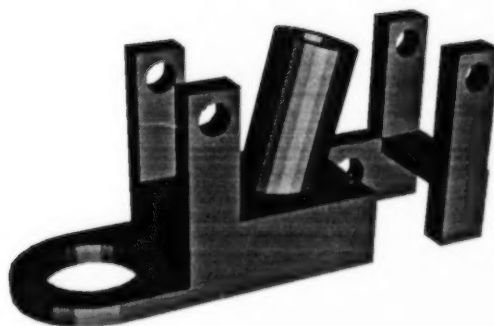


Figure 34: Test object B.

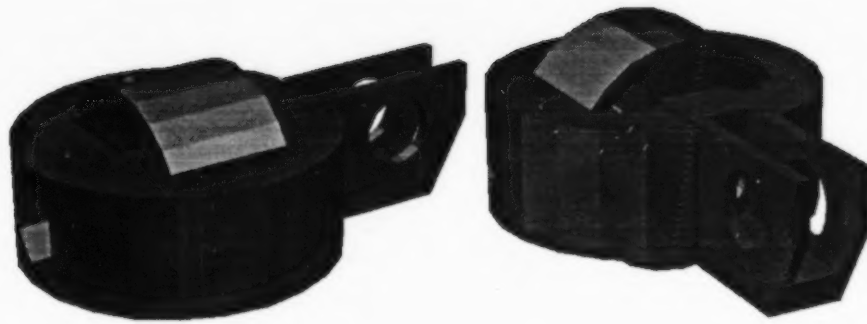


Figure 35: Test object C from two different views.

figure 36, 37 and 38).

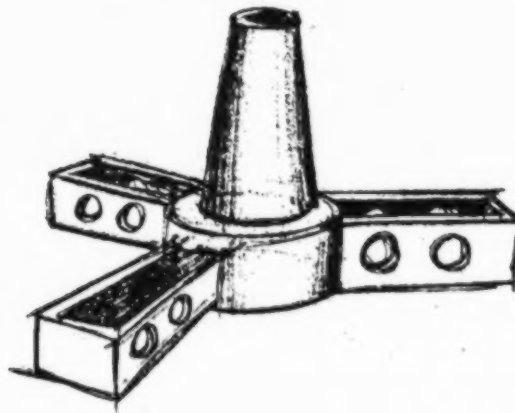


Figure 36: A sample sketch of the test object A.

8.2.3 Discussion of timing results

The following table shows the timing results of the sketching test. The time it took the author to create these test objects is also listed alongside for comparison.

Figure 39 shows a plot of the average times together with standard deviations.

As we can see from the timing data, creation time using JDCAD is shorter than the

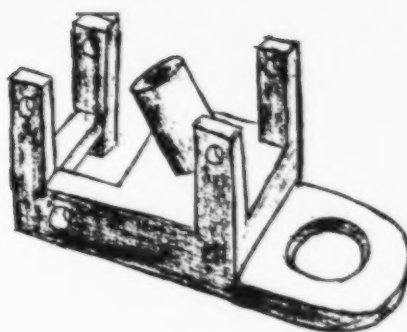


Figure 37: A sample sketch of the test object B.

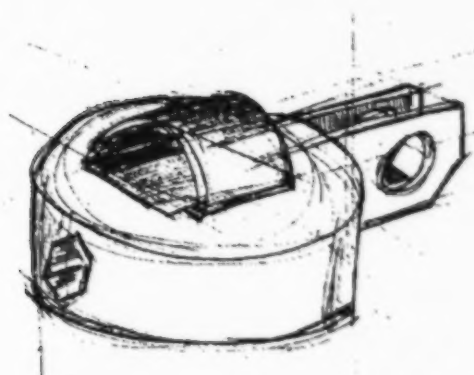


Figure 38: A sample sketch of the test object C.

Table 8.3: Hand sketching and JDCAD time (in seconds).

Object	Hand Sketching											JDCAD
	individual subject										avg.	
	1	2	3	4	5	6	7	8	9	10		
A	165	165	290	320	275	315	235	360	295	125	255	160
B	n/a	n/a	225	395	490	440	370	310	185	110	252	210
C	115	240	210	510	610	530	435	450	190	160	345	135

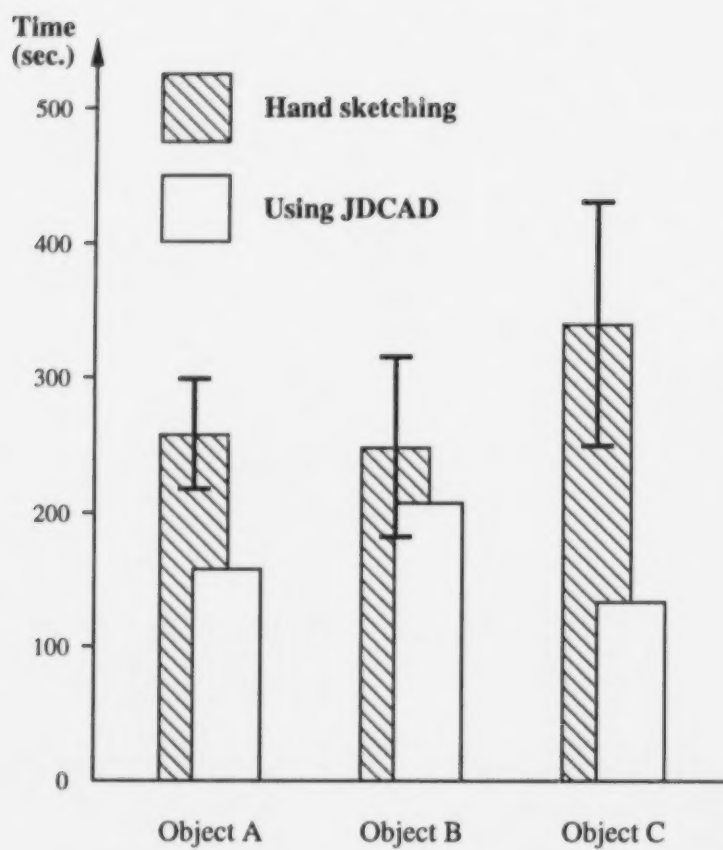


Figure 39: Comparison between hand sketching and using JDCAD.

average sketching time in all three cases. The *minimum* sketching time is also shorter than JDCAD time for all three objects. In general, JDCAD time is comparable with that of hand sketching.

To the author's best knowledge, this is the first reported comparison test between a CAD system and hand sketching. From the results in the previous section, it is not surprising that such test has not been conducted before, as common CAD systems are so helplessly slow compared with hand sketching that any actually comparison would be a waste of effort. To this extent, we have made significant progress in bringing CAD systems into the sketching stage.

As we can see from the timing data, there is a relative large amount of variation in sketching time. This is mainly due to the lack of standards for the quality of sketches. Sketching is not a science. Different people have different understanding as to what constitutes a "concept sketch containing enough detail to communicate a general idea about the 3D shape in question, but not too much details to fall behind the train of thought during conceptualization".

During the test, we observed that many subjects frequently refer back to the object displayed on screen, while a few of them mainly rely on memory. We asked several test objects about the possible effect of having an object displayed on screen. The answers were mixed. Two effects were identified. Some subjects suggested that having a "perfect" reference image tended to impede the sketching speed as one would want to match the high quality in the drawing process. Some subjects, on the other hand, suggested that such a reference image help speed up sketching as it provided a reference for curves (ellipses) that are difficult to draw otherwise.

It is interesting to compare the relative order of sketching time between two methods. For JDCAD, the order is $T_{\text{JDCAD}}(B) = 210 > T_{\text{JDCAD}}(A) = 160 > T_{\text{JDCAD}}(C) = 135$. For hand sketching, the order is exactly the reverse according to the averages, as $\bar{T}_{\text{Hand}}(B) = 252 < \bar{T}_{\text{Hand}}(A) = 255 < \bar{T}_{\text{Hand}}(C) = 345$. We conjecture that this is due to the characteristics of the shapes. Shapes that consist of mostly straight and parallel lines are relatively easy to draw by hand, since it is

easy to determine the directions and intersections of such lines. Shapes that contain curved lines or nonparallel lines require more time to draw by hand. Such effect does not apply to a CAD system, as geometric calculation usually takes a negligible amount of time compared with user interaction.

All the test subjects in this experiment were from a graduate level course in Industrial Design Studio and had experience in designing and sketching. When asked about comments, everyone pointed to the vital difference between a paper sketch and a complete 3D computer model. For example, one can turn a 3D model around and inspect it from different views, while it is impossible to do the same with a paper sketch. Many test subjects also commented on the effect of having a 3D model displayed on the screen while sketching. Such an image helped in (a) facilitating the drawing of projected curves (circular arcs), and (b) determining the intersection lines between different primitive shapes, especially when curved surfaces were involved.

The difference in sketching speed should not be taken at its face value, because there are very important differences between a paper sketch and a complete 3D computer model. Obviously, the value of the latter far exceeds that of the former.

The design of this experiment is not intended to account for all the confounding factors involved. For example, there is no commonly agreed-upon benchmark objects for this purposes. In fact, we do not even have a metric for shape complexity with respect to sketching, which is a very interesting research topic by itself.

8.2.4 Section summary

We compared the timing of hand sketching with that of sketching in JDCAD. We showed that a JDCAD expert can create the three test objects in more or less the same amount of time as with hand sketching. This is a very encouraging result, since it shows that even at the sketching stage, a computer can now compete with "used envelopes" [18]. Since most of the commercial CAD systems have yet to penetrate into the sketching stage in a designing process, due to the long turnaround time, we have in fact demonstrated the overall advantages of the interaction techniques in

JDCAD over those in other existing CAD systems.

8.3 Chapter summary

Two types of experiments have been carried out to evaluate the effectiveness of the new interface style.

First, we compared modeling time using JDCAD with those using a commercial CAD system, and showed that JDCAD is much faster than the commercial CAD system, in designing the two test objects.

Second, we compared the sketching time between using JDCAD and the traditional pencil-and-paper approach, and showed that a JDCAD expert can create the test objects using roughly the same amount time as hand sketching. Since the end result of using JDCAD is a complete 3D computer model, its usage for later design stages is far greater than that of a paper sketch. For this reason, we have achieved significant progress towards our ultimate goal of covering the sketching stage with CAD tools.

Chapter 9

Applications of JDCAD

9.1 VR demos

In the Fall semester of 1993, JDCAD+ (which is an extension from an earlier version of JDCAD), was used in the graduate level Computer Graphics course (CMPUT 511) as a tool for students to create virtual reality environments. In about two weeks time, most students learned to use JDCAD+, and built very interesting models with behaviors. Here we present some of them.

Figure 40 shows a dinosaur with some prehistoric plant. It is a snapshot of a student's assignment of using JDCAD+. The dinosaur has a walking behavior specified by keyframes. Figure 41 is taken from another student's assignment. It is one of two objects the student created for a virtual airport environment. The other object is a jeep (not shown). Figure 42 is taken from a similar assignment. The bug was assigned a walking motion in JDCAD+, and the knight is also assigned some arm movement. Figure 43 shows a helicopter designed in JDCAD+. It was first designed by hand coding vertices, and later redesigned when JDCAD+ became available [17].

As we can see from these examples, students with little or no background in 3D modeling can learn to build fairly interesting shapes in a short period of time. These models took just a few hours each to build. We observed that a large portion of the modeling time was actually spent on approximating a complex shape with a limited

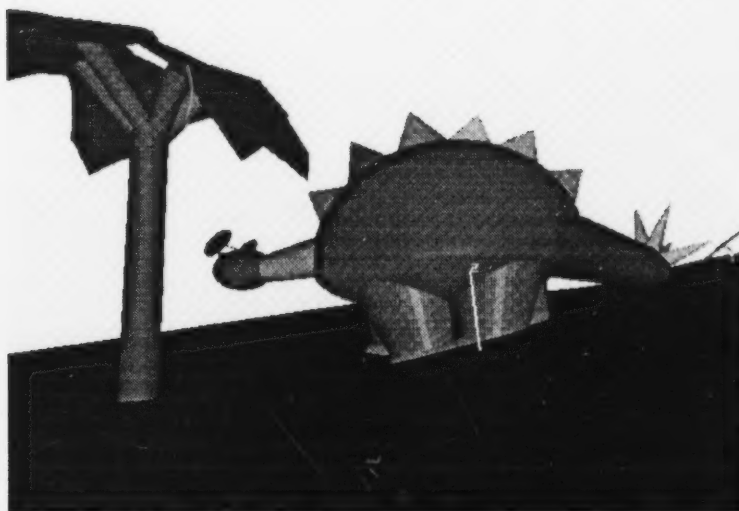


Figure 40: A snapshot from a dinosaur scene.

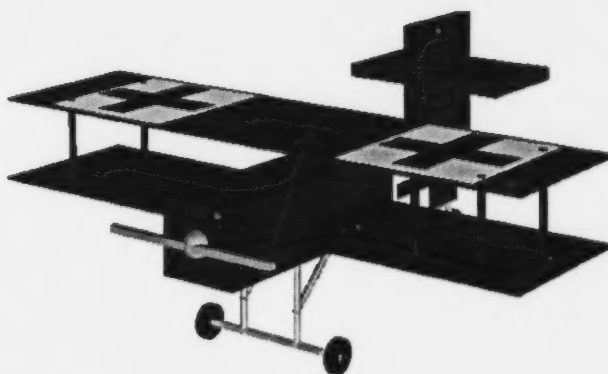


Figure 41: A biplane built in JDCAD+.

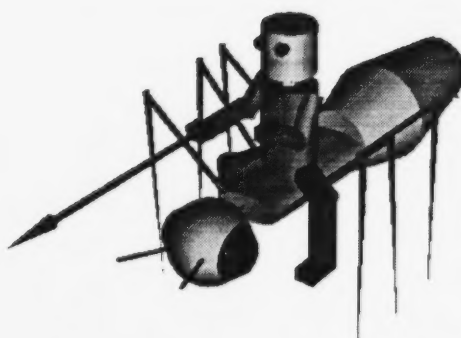


Figure 42: A knight riding a bug.

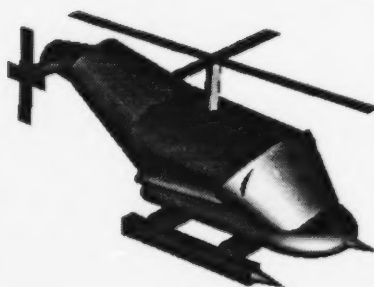


Figure 43: A helicopter designed in JDCAD+.

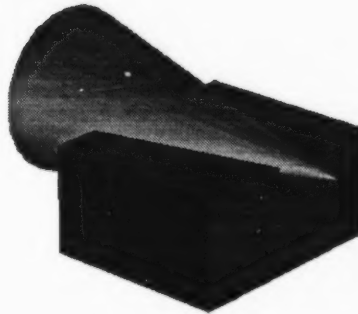


Figure 44: Illustrating a culling problem.

number of primitives, as constructive solid geometry operations were not available in JDCAD+. Had set operations been available, the modeling time would have been significantly lower.

9.2 Other applications

The intuitive interactions with 3D shapes also lend themselves to other applications, especially when the understanding of 3D spatial relationship is important but absolute precision is not required.

One day, several people of our graphics research group were discussing 3D culling algorithms. Not very knowledgeable about the problem, one person suggested to test all 8 vertices of an object's bounding box against the viewing cone as a visibility test. This criterion is insufficient, and could be illustrated by a hand drawing of a counterexample. However, as JDCAD happened to be running on one of the workstations at the time, one member simply used it to create a cone and a box, and placed the cone to intersect with the box but none of its vertices. This whole process took about 30 seconds, and illustrated the concept extremely well.

If one were to communicate such a concept by natural language, one would probably say something like: "Imagine you have a box in space, and a cone intersecting

one of its edge in the middle but leaving all the corners of the box outside, ..." The person at the receiving end would have to form a mental picture according to the verbal description, in order to understand the issue. It is usually not an easy task. Another alternative was to sketch a pictorial drawing on a piece of paper to visualize the situation. However, it would require some mental calculation to draw the intersection curves between the two objects to some precision in order to get the point across. It would not be an easy task either, just to consider that the intersection curves between the cone and the box in this situation are conic arcs, and they have to be projected through an oblique projection. Since this *ad hoc* use of JDCAD was so effective, the discussion went on with JDCAD as the primary supplement to verbal communication, replacing the old good pencil-on-paper method. This clearly demonstrates the advantage of 3D direct manipulation in communicating geometric concepts.

One test subject in the JDCAD versus AutoCAD comparison experiment is a school teacher. During the experiments, he envisioned the use of such a system in teaching solid geometry, and started to form models illustrating various spatial relationships (such as intersection curves between various shapes), that could be used in classes. In this case, the fact that one can manipulate the objects and get immediate feedback facilitates the understanding of spatial concepts and accelerates the learning process. Here, the advantage of using computer graphics techniques over using physical models becomes more evident, with the help of 3D direct manipulations. For example, building a computer model now takes much shorter time than crafting a physical one. Also the task of illustrating some spatial relationship regarding intersection seems to require an ideal "material" that is perfectly rigid and dense yet allows for free interpenetration. Only computer graphics coupled with 3D direct manipulation can put such an ideal matter at one's finger tip.

9.3 Chapter summary

While JDCAD was neither designed to be a full fledged 3D modeler, nor is it close to one now, it has been successfully used for several purposes, most importantly the building of virtual environments. It also showed great potential as a tool for communicating geometric concepts.

Chapter 10

Conclusions and Future Work

In this chapter, we first summarize this research, then discuss some topics for future research.

10.1 Summary

3D modeling is very important to computer graphics and computer aided design. Currently, it poses a major challenge, especially in the early stage of sketching. The main goal of this research is to explore ways of coping with the various types of difficulties involved. After making several observations about previous work and new technological advances, we came up with a new style. It was envisioned as a preferred interaction style for the sketching (or prototyping) of 3D objects, and one which can be seamlessly integrated into the entire design process. The main idea is to experiment with high DOF input devices in 3D direct manipulation, to gain insight into the unique characteristics of 3D interaction tasks, and eventually, to develop a good solution to the challenge.

In designing the overall style for visualization and kinaesthetic input, we tried to avoid pitfalls of two types of extremism in terms of interface design. The first is the conservative tendency to restrict oneself in using just traditional, low DOF input devices. The second is the radicalism in displacing nearly all the traditional input

devices with newly available, high DOF input devices and visualization schemes. Both types of approaches are faulty in their philosophies. Observing these facts, we strived for a harmonious congruence among all the techniques. We chose a non-immersive kinetic 3D display scheme to take advantage of the new tracking technology, while avoiding the undesirable side effects of immersive schemes.

We designed and implemented a prototype system, JDCAD, to gain real experience with such a style, and in turn, use the insights gained to further improve existing techniques and develop new ones. Many interaction techniques (such as menu selection and object selection) underwent many iterations of improvement. New aspects of the problem were revealed in the process, and they prompted the development of novel interaction techniques (such as the transparent rendering for previewing set operations).

The JDCAD system was also used to test out the effectiveness of the new style and associated interaction techniques. While there are many factors that make the results not as conclusive as one would want, they certainly show the great potential of the approach.

The major part of our experience with the JDCAD system can be summarized into the following guidelines.

First, use appropriate devices and their associated interaction techniques for tasks of different nature. For each specific task, one should identify the most appropriate input device, and develop related interaction techniques. So, one should not try to use a single type of device to cover all the tasks and displace all the rest. Instead, one should strive for a harmonic integration of all the available interaction techniques.

Second, design manipulation techniques to accommodate visualization, instead of bending visualization to suit manipulation. As noted by many researchers [41], understanding 3D structure is difficult, and we need the help of good visualization techniques. For example, occlusion is among the most important depth cues. However, rendering objects as opaque makes it difficult to carry out certain tasks, as occlusion can cause visibility problems. The multiple snap-grid technique and the re-

gion based reshaping technique are two examples of overcoming the difficulty caused by opaque rendering.

Third, avoid device switching by providing multiple entry points for each command. Most commands should be supported by as many input devices as possible, so the user is not forced to switch among input devices unless he or she feels that it is profitable to do so. Even with one particular input device, a task should be supported by multiple interaction techniques. While some techniques may be less than optimal for the task, it can save the time of switching among different input devices, and increase the overall efficiency. The ring menu is a good example of this principle — it is not the best menu selection technique by itself, but it saves the user from switching devices, thus becoming a technique favored by most JDCAD users.

These guidelines should be considered as rules of the thumb.

In the following section, we will discuss the intrinsic limitation of the style, and future research topics.

10.2 Limitations

During the period when various interaction techniques were developed and experimented with, several observations were made, which in turn inspired improvement over the techniques. Others revealed limitations which are intrinsic to the style.

There are several intrinsic limitations to our approach.

First, independent Cartesian control problem. Due to the nature of human motor system (muscle groups and joints), our hands do not have independent Cartesian control over 3D movement. As a result, simultaneous fine control in all three directions is difficult to achieve. This means that 3D free hand movement is suitable for coarse placement or reshaping, but not fine tuning, which requires the assistance of additional constraints, snap gridding, or text editing. While this problem exists even in 2D, adding one more dimension aggravates it. To minimize the effect caused by the lack of independent Cartesian control, one should supply a powerful yet flexible set of

constrained manipulations (e.g. constrained movement in a 1D and 2D). Supplying a naturally constrained input, such as mouse or tablet input, would also be helpful.

Second, open loop problem. When one's hand is unsupported, the motor control loop becomes open, as there is no force feedback for hand movement. This decreases the accuracy in movement control and increases the transient time. Since no device today offers unencumbered 3D input together with force feedback, this is the price we have to pay in order to get full 3D input. This fact again emphasizes the importance of traditional input devices such as mice and tablets.

Third, stability problem. This is mainly due to the lack of force feedback. When one's hand is supported only at the elbow, the stability of the hand is limited. Physical fatigue and other facts can cause the position of the bat to wander around over time. The amplitude is usually at a few millimeters. Due to this factor, fine control at millimeter level is difficult. This is a new problem introduced by the bat, and it does not exist in the traditional 2D style.

Fourth, device imperfection. Since the Polhemus Isotrak uses electromagnetic field for sensing, it is subject to interference from a number of sources. For example, any metal objects may distort the electromagnetic field and cause space warping. Other electric and electronic devices such as tablets and CRT terminals may also cause and/or receive interference. All these contribute to the noise in measurement data and hinder user interaction.

10.3 Future topics

Based on the experience we obtained, a better interaction style can be outlined.

First, there should be a balance between low degrees of freedom input and high degrees of freedom input. For the sake of convenience, we will use "2D" to represent input up to two degrees of freedom (such as permitted by a mouse or tablet), and use "3D" to represent input with three or more degrees of freedom. Given the fact that over half of the design activities are 2D in nature, a traditional 2D input device should

be supplied to facilitate them. Also, since a 3D input device can save a significant amount of time when it comes to 3D operations (such as kinetic 3D visualization, viewing specification, interactive interrogation, and so on), a 3D input device is also necessary. So it comes down to a suitable way of combining these two types of devices into one integrated device. One possible solution is to attach a 3D sensor onto a tablet-based stylus. When the tip of the stylus is on the tablet surface, the computer takes its input from the 2D tablet just as an ordinary 2D input device. When the stylus is lifted from the tablet surface, the computer takes its input from the location and orientation from the 3D sensor. In this way, the designer will have easy access to both 2D and 3D input at the same time. If the tablet and 3D sensor do not interfere with each other, then the designer will also have one rotational degree of freedom around the length of the stylus when the tip is in contact with the surface. This will further empower the designer by a true rotational input, which is lacking for traditional 2D input device.

This new style will combine the advantages of both 2D and 3D devices and eliminate most of their disadvantages. This will further reduce the prototyping time. It will also help further reduce physical fatigue. Of course, such a hybrid system also requires a way of making a smooth transition from 2D-based operations to 3D-based ones.

10.4 Other related topics

There are other design issues that are not adequately addressed in this research, and merit further consideration.

First, geometric constraints. One important factor that makes traditional sketching very useful is its inherent ability for maintaining certain geometric constraints [42]. Due to limited resources, we are unable to incorporate automatic constraint satisfaction into our prototype system. However, our experience with JDCAD so far clearly indicates the importance of constraints. The specification and visualization of

geometric constraints in a 3D environment is certainly a challenging and intriguing topic for further research.

Second, freeform surfaces. Among various branches of computer aided design, the one with the largest amount of theoretical results is probably the design of freeform surfaces. Nevertheless, the actual practice of freeform design is still largely difficult and time consuming. One of the major problems here is the vast discrepancy between the input and the object. A traditional input device has only up to two degrees of freedom, however, a freeform surface (at least in theory) may have infinite degrees of freedom. Even if we limited the degrees of freedom by subjecting the surface to mathematical formula (e.g. a tensor product spline form), the resulting degrees of freedom could still be quite large, making it difficult to manipulate using a 2D input. With the additional degrees of freedom available with our interaction style, this problem may be tackled with a better method.

Third *seashell* mode versus *scuba* mode. As a general practice in engineering design, most designs are sketched and drafted in a scale fit for easy reading and handling. At the sketching stage, a design is usually at the note pad size. This is sometimes called the seashell mode, as the model being designed is scaled to roughly the size of a seashell that can be comfortably held in one's hand. However, there are cases where a scaling may not be desirable, as it may alter the perception of certain aspects of the model. For example, a virtual walkthrough of an architectural design has to be in its true scale in order to fully reveal its spatial aspect. A slight scaling up would give a false sense of spaciousness, and slight scaling down will do the reverse. We call it the scuba mode when the designer is placed in an immersive environment and its scale is important. Our interaction style puts the designer in the seashell mode. As a result, it may not be suitable for design tasks that require the scuba mode. A smooth transition between the seashell mode and the scuba mode warrants further research.

Recently, Stoakley et al [41] developed the technique World in Miniature (WIM), which allows the user to view a model at different scales simultaneously. It is a

promising way of resolving the above mentioned conflict between seashell mode and scuba mode.

10.5 Concluding remarks

Even though we have gained much insight into the problem of 3D rapid prototyping and 3D interactions in general, we are still far from understanding them as well as we do in the 2D realm. However, we have made significant progress in giving a positive answer to the question "Can computers compete with used envelopes" in quick designing of 3D models. Although hand sketching still has many advantages such as low cost, light weight, and high flexibility, the final result from a CAD modeler is more useful than a paper sketch for many purposes. The new interaction techniques together with the guidelines derived from this dissertation research should provide a foothold towards an ideal medium for 3D modeling.

Appendix A

JDCAD

A.1 Introduction to JDCAD

JDCAD is a highly interactive 3 dimensional geometric modeling system. It allows the user to interact with a 3D model directly *in* 3D, by using a button-enhanced 3D tracker device (the bat). The user can create, place, or modify 3D shapes by a combination of keyboard commands, button actions, and bat movements. 3D shapes are usually displayed as solids in a perspective view, with real time updates corresponding to user manipulations. Various interaction techniques facilitate the modeling task, especially for quick sketching. We give a brief introduction to the concepts and terminologies before we list the commands.

A.1.1 Object type and hierachy

A 3D model consists of one or more objects. Each object is either a solid or a collection of objects. A solid is a primitive shape, or the result of a set operation on two solids. A collection of objects is formed by a "group" operation on objects.

A.1.2 States and roles

An object can be in one of two states: normal or selected.

There are three roles that a selected object can play: primary, secondary, and tertiary. When only one object is selected, it automatically becomes a primary selection. When two objects are selected, the one that is selected first becomes the primary selection, and the other secondary selection. All the objects that are selected after the second one become tertiary selections. The role of a selected object can be altered by user commands.

Objects in normal state appear grey. Objects in selected state are high lighted with color. The primary selection appears red, the secondary selection blue, and the tertiary ones green.

The primary selection is the target of most operations, such as translation, rotation, alignment, conformation, and reshaping.

The secondary selection serves as a reference frame for alignment operations and also as the second operand in a set operation.

The tertiary selection only participates in a group operation to form a group object.

A.1.3 Active axis

A user can activate one of the three axes to constraint certain operations. When no object is selected, the axes available for activation are those of the modeling coordinate system. If there is only a primary selection, the axes available for activation are those of the local coordinate system of the primary object. When there is a secondary selection, the axes of its local coordinate system are available for activation.

A.2 List of Commands

Most commands in JDCAD are single key presses. This is mainly because of speed considerations. Single key presses are designed to be executed by the left hand, and they save the time for switching the right hand between the bat and the keyboard.

Commands that do not require bat motion (for instance, set operations, state

toggles) are executed upon key presses. Commands that require bat motion (for instance, rubber-banding, dragging) are activated upon key presses and finalized upon key releases.

A.2.1 Creation of primitive shapes

Currently, JDCAD supplies nine types of primitive shapes: box, cylinder, cone, truncated cone, sphere, hemisphere, hexagonal prism, right triangular prism, and right tetrahedron.

- <F2>** Brings up the ring menu to set the current primitive shape.
- <F3>** Creates an instance of the current primitive shape by 3D rubber-banding.

A.2.2 Selecting objects

The selection state of an object can be changed either by using the spotlight selection or by keyboard commands.

- s** Turns on the spotlight. Releasing the key selects the closet object.
- S** Toggles the selection state of the closest object. Used for multiple selection or de-selection.
- e or Alt-a** Selects all the objects in the environment.
- w** Circles the primary selection in the selection set.
- W** Circles the primary selection in the selection set in a reverse order.
- Arrow keys:** **<Up>**, **<Down>**, **<Left>**, **<Right>** Moves the primary selection focus in the object hierarchy. **<Up>**: parent; **<Down>**: first child; **<Left>**: left sibling; **<Right>**: right sibling.

A.2.3 Selecting axis

The bat is used to select the axis to be activated or deactivated. The user only needs to point the bat at roughly the direction along the axis, and the system finds out the closest axis.

a Toggles an axis between active and non-active.

A Deactivate an axis.

A.2.4 Placement of objects

The primary selection can be directly placed with the bat, subject to possible axial constraints and snapping.

If there is an active axis, the translation of the object is constrained to be along the axis. Otherwise, translation in all three dimensions can be effected simultaneously. When translation snap is on, dynamic snapping occurs when certain conditions are met. See section 7.2.1 for details.

The rotation of the primary object is controlled differently than that of translation. When there is an active axis, rotation is constrained to be around the axis. When rotation snap is on, the rotation angle is snapped to be multiples of 15 degrees. Otherwise, it is continuous. If there is no active axis, then rotation is snapped to a multiple of 90 degrees around any axis when rotation snap is on; and completely unconstrained otherwise.

t Translates the primary object with the bat.

r Rotates the primary object with the bat.

g Translations and rotates the primary object.

⓪ Toggles rotation snap mode.

Ctl1-⓪ Toggles translation snap mode.

A.2.5 Transformation of model

The entire model can be manipulated by the bat in a similar way an object is manipulated.

- Alt-t** Translates the environment with the bat.
- Alt-r** Rotates the environment with the bat. The center of rotation is the center of the design space. <F1> has the same function.
- Alt-g** Translations and rotates the environment. **Shift-<F1>** has the same function.

A.2.6 Reshaping of objects

An object can be reshaped by freehand rubber-banding or by precise textual editing. The former is meant for quick sketching, while the latter for detailing.

- Alt-s** Reshapes a primitive object. The nature of the reshaping operation is determined by the starting position of the 3D cursor with respect to the object — so-called region-based reshaping.

For a cylinder object, **Ctl-1** (one) toggles the cursor between a jack (the normal shape) and a reshaping indicator (region sensitive cursors). For a cone, starting the reshaping with the cursor slightly inside the pinnacle brings up the dial to adjust spread angle. Pressing **1** (one) when **Alt-s** is pressed brings up a finer dial for better control. When reshaping a box, three operations are available: pulling a face, an edge, or a corner.

- Alt-e** Pops a dialog box for text-based editing of a primitive object.

A.2.7 Alignment and conformation

An alignment or conformation operation acts on the primary object (which, in the case of a conformation operation, has to be an primitive object, not a composite one),

using the secondary object as reference.

- 1 Axial alignment lower-end to lower-end.
- 2 Axial alignment mid to mid.
- 3 Axial alignment upper-end to upper-end.
- 4 Axial alignment upper-end to lower-end.
- 5 Axial alignment lower-end to upper-end.
- 6 Axial alignment upper-end to mid.
- 7 Axial alignment lower-end to mid.
- 8 Axial alignment mid to lower-end.
- 9 Axial alignment mid to upper-end.
- 0 Concentric alignment around the current axis.
- Parallel alignment. No axis required.
- + Makes coordinate systems aligned and conforms the primary objects to the secondary in all directions.
- # Center align the primary object to the secondary object. No axis required.
- [or = Full-size conformation along the current axis.
-] Half-size conformation along the current axis.

A.2.8 Set and group operations

A set operation needs two operands: the primary (red) object and the secondary (blue) object. A group operation involves all selected objects (two or more). After a set/group operation is executed, its result is displayed in the place of its participating objects. These participating objects are no longer available for direct manipulation. However, a user can "open" up a composite object to reveal its components and manipulate them. The object hierarchy is maintained during these manipulations, and when the user "closes" the object, the new result of the original set/group operation is computed and displayed.

- u Unions the two objects.
- i Intersects the two objects.
- o Subtracts the secondary from the primary (Red - Blue).
- p Subtracts the primary from the secondary (Blue - Red).
- \$ Group the selected objects.
- % Ungroup the selected object.
- <F4> Toggles transparent rendering mode. Only affects the red and blue objects.
- (Opens the set/group node for manipulation of its components. <Tab> has the same function.
-) Closes the opened node. (Performing the node function again on possibly changed components). Shift-<Tab> has the same function.
- * Opens up all the nodes starting from the primary selection.

A.2.9 Cursor control

The mapping from the physical location of the bat to the 3D cursor in the design space can be dynamically changed by keyboard commands.

- ' Homes the cursor to the center of the room.
- ~ Homes the cursor to the center of the screen.
- <Esc> Clutching. Pressing <Esc> disengages the cursor from the bat. Releasing <Esc> re-engages it.

A.2.10 Viewing and rendering

Viewing and display modes can be controlled from keyboard as well as from pull-down menus. A spacemark is a stored view. A user can set multiple spacemarks and recall them by traversing up and down the chain of spacemarks.

- z Zooming out (objects will appear smaller).
- Z Zooming in (objects will appear larger).
- Ctl-z Moves the primary object into the center of the room.
- Alt-z Rotates the environment so that the current object is aligned with the room.
- <Return> Set the current view as a spacemark.
- Shift-<Return> Go to the last spacemark.
- <PgUp> Go to the previous spacemark.
- <PgDn> Go to the next spacemark.
- Alt-f Flies the user along the direction of the bat.
- Alt-b Flies the user along the direction of the bat.

- <F6> Toggles between solid and wireframe rendering.
- <F8> Toggles between perspective projection and parallel projection.
- Alt-c Circles among four clipping modes: no clipping, half space, quarter space, octant space.

A.2.11 General editing

JDCAD provides a clipboard for general editing of objects.

- c Copies the primary object to the clipboard.
- C Clears out the primary object. Ctl-c has the same function.
- v Pastes the object on the clipboard.
- d Duplicates the primary object. Ctl-d has the same function.
- x Cuts the primary object and copies it to the clipboard.

A.2.12 File operations

JDCAD has its own file format for 3D models. It stores the object hierarchy in a tree-like structure. Set operations are evaluated upon loading.

- Ctl-q Quits without saving.
- Ctl-s Saves into file.
- Ctl-a Saves into a new file.
- Ctl-l Loads from a file after clearing the current model.
- Ctl-p Loads from a file without clearing the current model.
- Ctl-n Clears the current model.

A.2.13 Buttons

Three buttons are attached to a Polhemus Isotrak sensor to form a bat. These three buttons are named the nose button, the dorsal button and the tail button according to their position on the sensor casing. The nose button is used to create a primitive shape. Its function is the same as the <F3> key. The dorsal button is used to move an object. Its function is the same as the **t** key. The tail button is used for rotating the environment, same as **Alt-r** and **shift-<F1>**.

A.2.14 Pull-down menus

Pull-down menus provide some functions inaccessible from the keyboard. For example, turning the room rendering on or off, changing viewing control (head, mouse, static), etc.

A.2.15 Miscellaneous

<F11> Take snapshot of the screen, and dump the image into file **scene.pgm** in PGM (portable grayscale map) format. **Alt-2** has the same function.

A.3 Summary

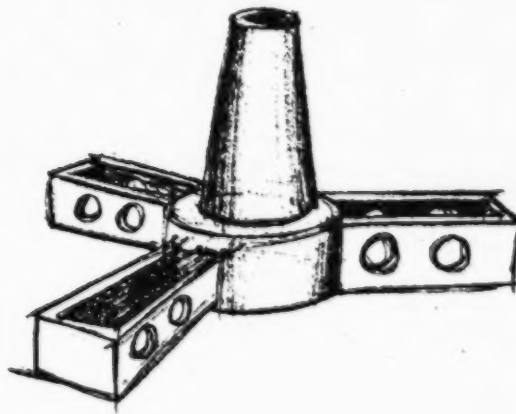
We present a brief introduction to the JDCAD system. It is meant to be a prototype and demo-of-concept, instead of a full blown CAD package. Most of its functionalities are developed to facilitate quick sketching.

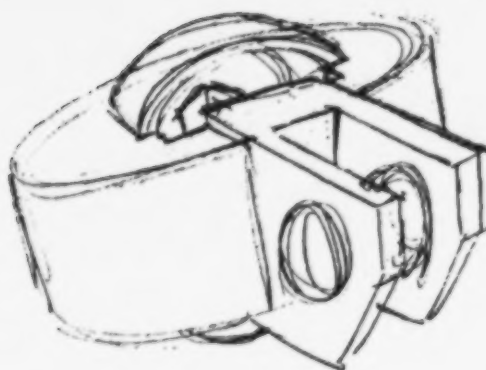
Appendix B

Hand Sketches

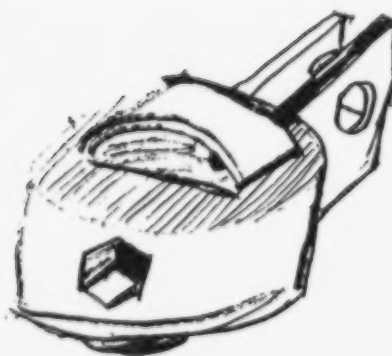
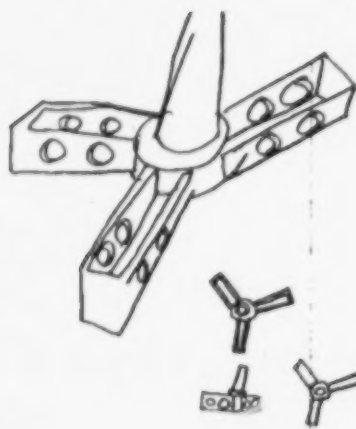
We include all the hand sketches produced in the experiment. They were drawn with pencil on paper and then scanned into black and white images. They are in either 1:2 or 1:4 scale to the originals.

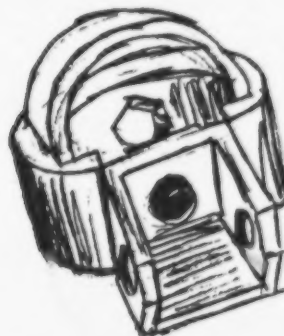
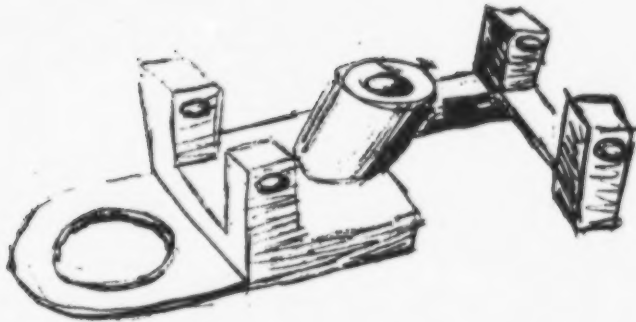
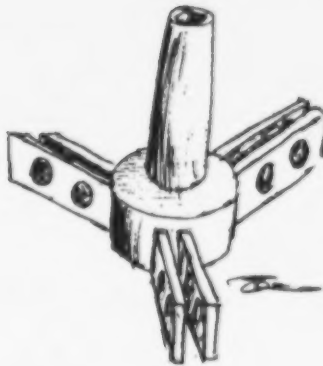
B.1 Subject No. 1.



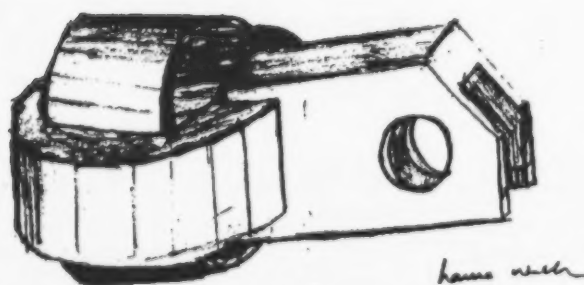
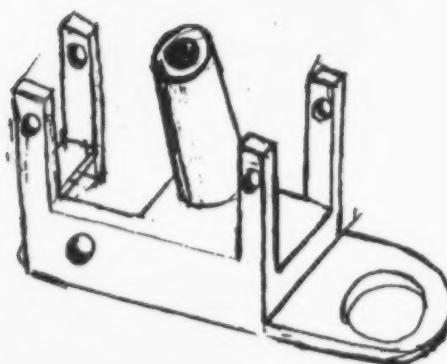
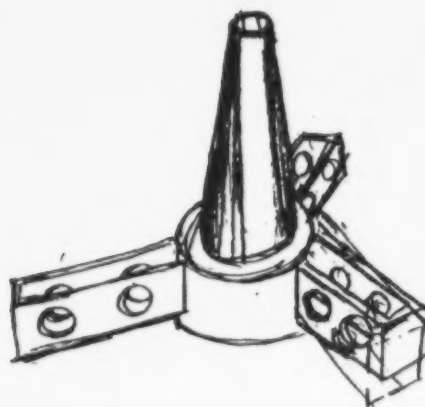


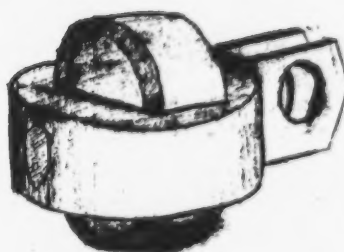
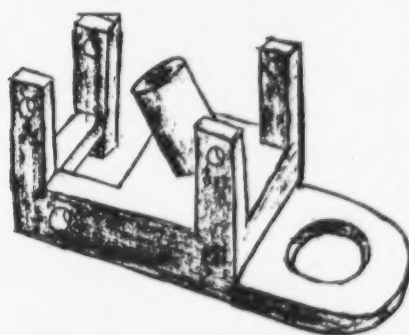
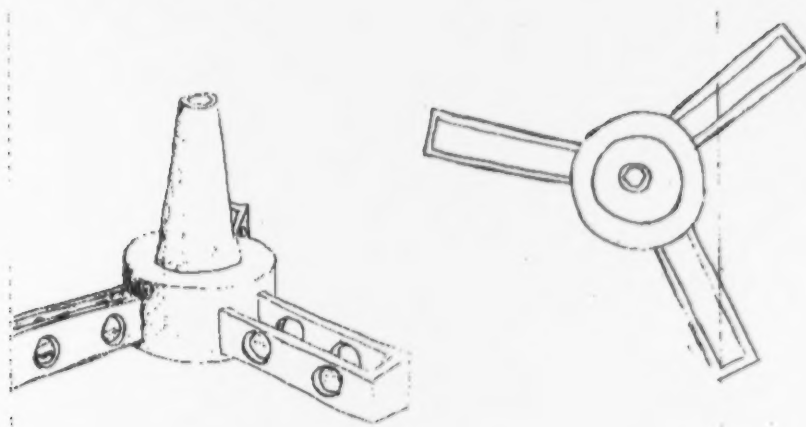
B.2 Subject No. 2.



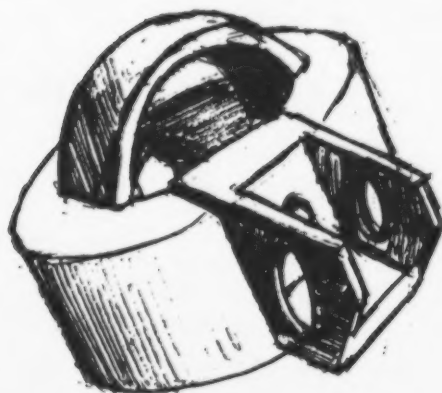
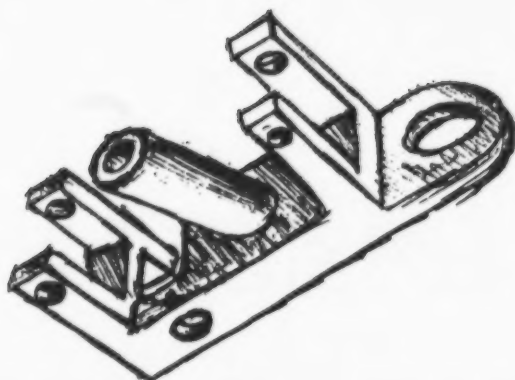
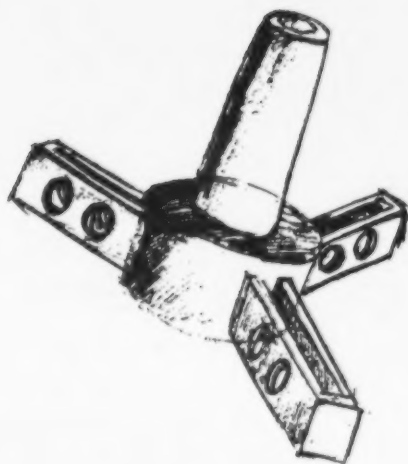
B.3 Subject No. 3.

B.4 Subject No. 4.

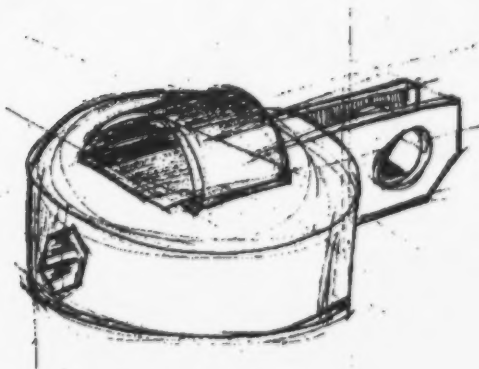
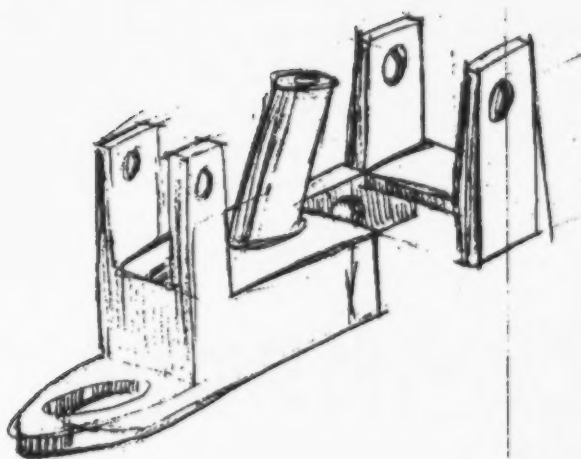
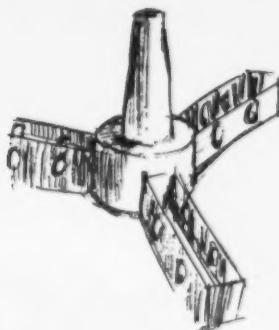


B.5 Subject No. 5.

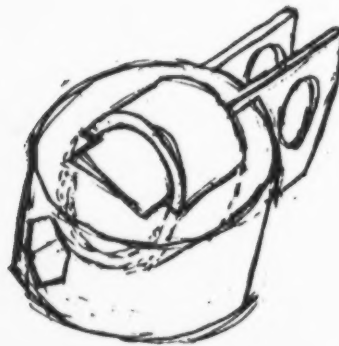
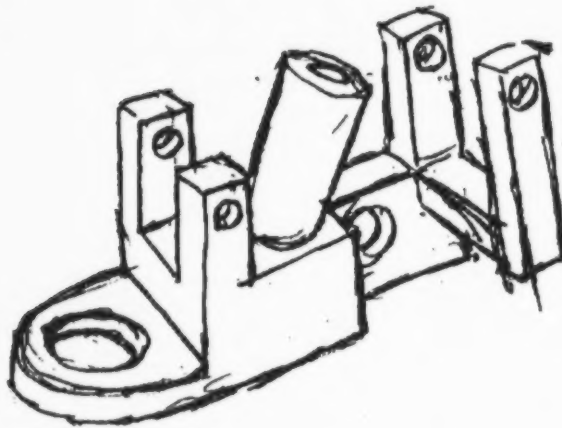
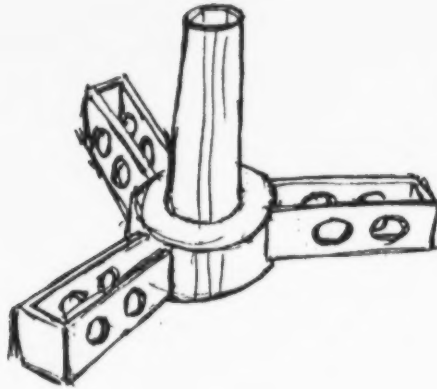
B.6 Subject No. 6.



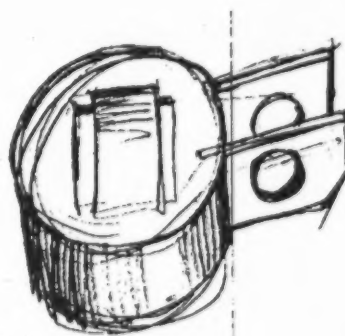
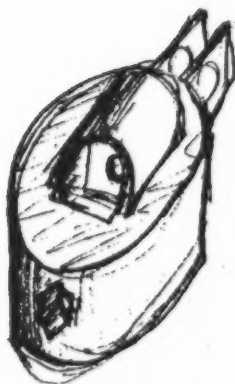
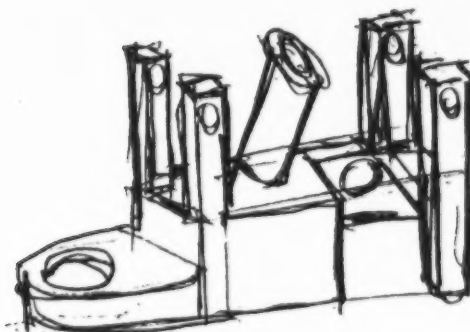
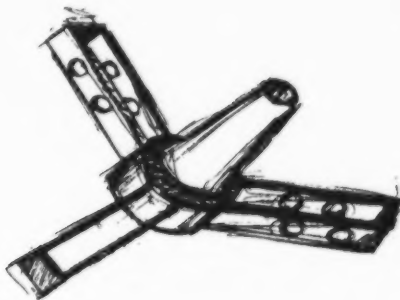
B.7 Subject No. 7.

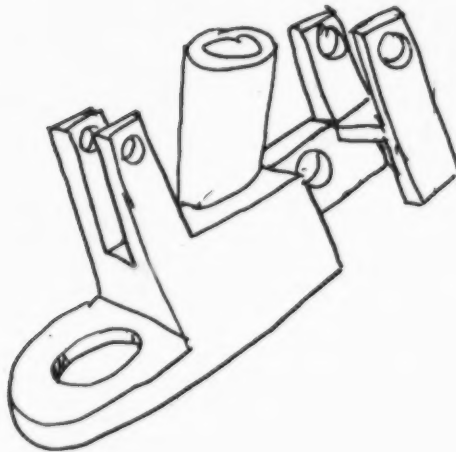
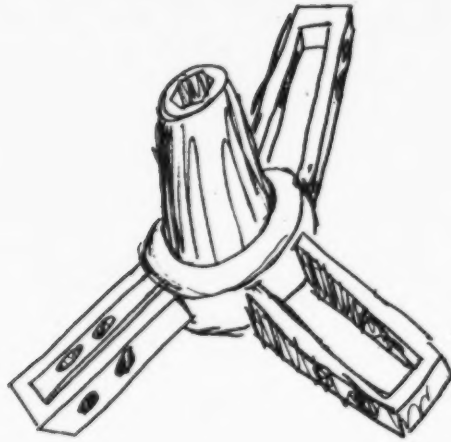


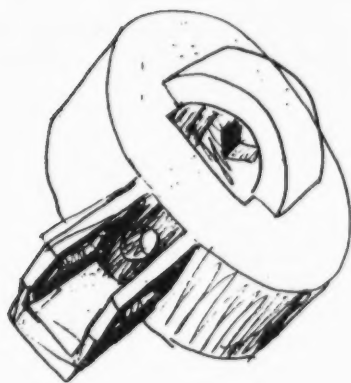
B.8 Subject No. 8.



B.9 Subject No. 9.



B.10 Subject No. 10.



Bibliography

- [1] Makoto Akeo, Hiroshi Hashimoto, Taisuke Kobayashi, and Tetsuo Shibusawa. Computer Graphics System for Reproducing Three-Dimensional Shape from Idea Sketch. In M. Daehlen and L. Kjeldahl, editors, *EUROGRAPHICS '94*, volume 13, pages 477-488. Eurographics Association, Blackwell Publishers, 1994.
- [2] Hossein Anani and Reza Khorramshahgol. The impact of Automation on Engineers' Creativity and Innovation and Its Implications for Reducing Resistance to Change. *Computers in Industry*, 16:377-383, 1991.
- [3] Kevin Arthur. 3D Task Performance in Fish Tank Virtual Reality. In *Proc. 5th Annual Western Computer Graphics Symposium*, Vernon, British Columbia, March 1993.
- [4] Norman I. Badler, Kamarn H. Manoochehri, and David Baraff. Multi-Dimensional Input Techniques and Articulated Figure Positioning by Multiple Constraints. In *Proc. 1986 Workshop on Interactive 3D Graphics*, pages 151-169, Chapel Hill, North Carolina, October 1986.
- [5] Eric A. Bier. Snap-Dragging in Three Dimensions. In *Proc. 1990 Symposium on Interactive 3D Graphics*, volume 24, pages 193-204, Snowbird, Utah, March 1990.
- [6] Eric Allan Bier. Skitters and Jacks: Interactive 3D Positioning Tools. In *Proc. 1986 Workshop on Interactive 3D Graphics*, pages 183-196, Chapel Hill, North Carolina, October 1986.
- [7] Vasco Branco, Antonio Costa, and F. Nunes Ferreira. Sketching 3D Models with 2D Interaction Devices. In M. Daehlen and L. Kjeldahl, editors, *EUROGRAPHICS '94*, volume 13, pages 489-502. Eurographics Association, Blackwell Publishers, 1994.
- [8] Steve Bryson. Paradigms for the Shaping of Surfaces in a Virtual Environment. ACM SIGGRAPH '92 Course Notes, No. 9, July 1992.
- [9] J. Butterworth, A. Davidson, S Hench, and T. M. Olano. 3DM: A Three Dimensional Modeler Using a Head-Mounted Display. In *Proc. 1992 Symposium on Interactive 3D Graphics*, pages 135-138, Cambridge, Massachusetts, March 1992.

- [10] Stuart K. Card, Thomas P. Moran, and Allen Newell. *The Psychology of Human-Computer Interaction*. Lawrence Erlbaum Associates, Inc., 365 Broadway, Hillsdale, New Jersey 07642, 1983.
- [11] James H. Clark. Designing Surfaces in 3-D. *Communications of the ACM*, 19(8):454-460, August 1976.
- [12] D. Brookshire Conner and et al. Three-Dimensional Widgets. In *Proc. 1992 Symposium on Interactive 3D Graphics*, pages 183-188, March 1992.
- [13] Michael Deering. High Resolution Virtual Reality. In *Proc. ACM SIGGRAPH '92*, pages 195-202, Chicago, Illinois, July 1992.
- [14] J. Encarnacao. CAD Technology: A Survey on CAD Systems and Applications. *Computers in Industry*, 8:145-150, 1987.
- [15] Julian E. Gomez et al. Why is 3-D Interaction So Hard and What Can We Really Do About It? In *Proceedings of SIGGRAPH 1994*, pages 492-493, Reading, MA, July 1994. Addison-Wesley Publishing Company.
- [16] William Fitzgerald, Franklin Gracer, and Robert Wolfe. GRIN: Interactive Graphics for Modeling Solids. *IBM Journal of Research and Development*, 25(4):281-294, July 1981.
- [17] Sean Haliday and Mark Green. A Geometric Modeling and Animation System for Virtual Reality. In Gurminder Singh, Steven K Feiner, and Daniel Thalmann, editors, *Proceedings of 1994 Symposium on Virtual Reality Software and Technology (VRST '94)*, pages 71-84, Singapore, August 1994. World Scientific.
- [18] Jozef Hatvany. Can Computers Compete with Used Envelopes. In *Proc. IFAC 9th Triennial World Congress*, pages 3373-3375, Budapest, Hungary, 1984.
- [19] Ken Hinckley, Randy Pausch, John C. Goble, and Neal F. Kassell. A Survey of Design Issues in Spatial Input. In *Proc. ACM Symposium on User Interface Software and Technology*, pages 213-222, Marina del Rey, California, November 1994.
- [20] M. Hodkinson and R. Patel. 3-D Modeling in a Teaching Company Scheme. *Computer-Aided Engineering Journal*, pages 177-180, October 1989.
- [21] Robert Jacobson. CAD Off the Screen: The Virtual World as a More Natural Design Environment for Architects. In *Proc. ACST '93*, Wiesbaden, Germany, November 1993.
- [22] Frederick P. Brooks Jr. Grasping Reality Through Illusion — Interactive Graphics Serving Science. In *CHI '88 Conference Proceedings Human Factors in Computing Systems*, pages 1-11, Washington D.C., May 1988.

- [23] Y. E. Kalay. Worldview: An Integrated Geometric-Modeling/Drafting System. *IEEE Computer Graphics and Applications*, 7:36-46, January 1987.
- [24] T Kuragano. FRES DAM System for Design of Aesthetically Pleasing Free-Form Objects and Generation of Collision-Free Tool Paths. *Computer-Aided Design*, 24(11):573-581, November 1992.
- [25] Fred Lakin, John Wambaugh, Larry Leifer, Dave Cannon, and Cecilia Sivard. The Electronic Design Notebook: Performing Medium and Processing Medium. *The Visual Computer*, 5:214-226, 1989.
- [26] Jiandong Liang and Mark Green. Quick Prototyping for Engineering Design. In *Proc. ASME Symposium on Computers in Engineering*, Houston, Texas, January 1993.
- [27] Jiandong Liang, Chris D. Shaw, and Mark Green. On Temporal-Spatial Realism in the Virtual Reality Environment. In *Proc. ACM Symposium on User Interface Software and Technology*, pages 19-25, Hilton Head, South Carolina, November 1991.
- [28] W. Myers. An Industrial Perspective on Solid Modeling. *IEEE Computer Graphics and Applications*, 2:86-97, March 1982.
- [29] Gregory M. Nielson. CAGD's Top Ten: What to Watch. *IEEE Computer Graphics and Applications*, 13(1):35-37, January 1993.
- [30] Gregory M. Nielson and Dan R. Olsen Jr. Direct Manipulation Techniques for 3D Objects Using 2D Locator Devices. In *Proc. 1986 Workshop on Interactive 3D Graphics*, pages 175-182, Chapel Hill, North Carolina, October 1986.
- [31] George G. Robertson, Stuart K. Card, and Jock D. Mackinlay. The Cognitive Coprocessor Architecture For Interactive User Interfaces. In *Proc. ACM Symposium on User Interface Software and Technology*, November 1989.
- [32] Jarek Rossignac, Abe Megahed, and Bengt-Olaf Schneider. Interactive Inspection of Solids Cross-sections and Interferences,. In *Computer Graphics*, volume 26, pages 353-360, SIGGRAPH '92, July 1992.
- [33] Jaroslaw R. Rossignac. Constraints in Constructive Solid Geometry. In *Proc. 1986 Workshop on Interactive 3D Graphics*, pages 93-110, Chapel Hill, North Carolina, October 1986.
- [34] Emanuel Sachs, Andrew Robers, and David Stoops. 3-Draw: A Tool for designing 3D Shapes. *IEEE Computer Graphics and Applications*, 11:18-24, November 1991.

- [35] Chris D. Shaw, Mark Green, Jiandong Liang, and Yunqi Sun. Decoupled Simulation in Virtual Reality with the MR Toolkit. *ACM Transactions on Information Systems*, 11(3):287-317, July 1993.
- [36] Chris D. Shaw, Jiandong Liang, Yunqi Sun, and Mark Green. The Decoupled Simulation Model for Virtual Reality Systems. In *Proc. CHI '92 Conference on Human Factors in Computing Systems*, pages 321-328, Monterey, California, March 1992.
- [37] C. Shmandt. Spatial Input/Output Correspondence in a Stereoscopic Computer Graphics Work Station. *Computer Graphics*, 17(3):253-261, July 1983.
- [38] Ken Shoemake. ARCBALL: A User Interface for Specifying Three-Dimensional Orientation Using a Mouse. In *Proceedings Graphics Interface '92*, pages 151-156, Vancouver, British Columbia, Canada, May 1992.
- [39] E. Sittas. 3D Design Reference Framework. *Computer-Aided Design*, 23(5):380-384, 1991.
- [40] R. F. Sproull. Parts of the Frontier Are Hard to Move. In *Proc. ACM SIGGRAPH 1990*, March 1990.
- [41] Richard Stoakley, Matthew J. Conway, and Randy Pausch. Virtual Reality on a WIM: Interactive World in Miniature. In *Proc. CHI '95 Conference on Human Factors in Computing Systems*, pages 265-272, Denver, Colorado, May 1995.
- [42] David G. Ullman, Stephen Wood, and David Craig. The Importance of Drawing in the Mechanical Design Process. *Computer Graphics*, 14(2):263-274, 1990.
- [43] M. J. G. Maarten van Emmerik. A Direct Manipulation Technique for Specifying 3D Object Transformation with a 2D Input Device. *Computer Graphics Forum*, 9:335-361, 1990.
- [44] Colin Ware, Kevin Arthur, and Kellogg S. Booth. Fish Tank Virtual Reality. In *Proc. INTERCHI '93*, pages 37-41, Amsterdam, The Netherlands, April 1993.
- [45] Colin Ware and Danny R. Jessome. Using the Bat: A Six Dimensional Mouse for Object Placement. In *Proc. Graphics Interface '88*, pages 6-10, Edmonton, Alberta, June 1988.